Lehrstuhl für Integrierte Schaltungen
Technische Universität München

Low Power ASIC Design
Using Voltage Scaling at the Logic Level

Torsten Mahnke

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzende: Univ.-Prof. Dr. rer. nat. Doris Schmitt-Landsiedel

Prüfer der Dissertation:

1. Univ.-Prof. Dr.-Ing. Ingolf Ruge, em.
2. Univ.-Prof. Dr.-Ing. Hans-Jörg Pfleiderer, Universität Ulm

Diese Dissertation wurde am 19.05.2003 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 11.12.2003 angenommen.

For Rainer,
who has been a good friend
ever since we were kids.

Shirts in the closet,
shoes in the hall
Mama's in the kitchen,
baby and all
Everything is everything
Everything is everything
But you're missing

(Bruce Springsteen)

Acknowledgements

First of all, I would like to thank Prof. Dr. Ingolf Ruge and Dr. Walter Stechele who enabled me to pursue this dissertation at the Institute for Integrated Circuits. Moreover, Dr. Walter Stechele deserves special thanks for his encouragement at particularly difficult stages of my work. I am also very grateful to Prof. Dr. Hans-Jörg Pfleiderer for his interest in this work and for serving on the dissertation committee.

I am particularly grateful to those colleagues and friends of mine who spent their time proof-reading the manuscript of this thesis. In alphabetical order, these are Dr. Ralf Altherr, Stephan Henzler, Prof. Dr. Richard Hornsey, Dr. Robert Klinski, Herbert Kolaric, Chris Menkus, Dr. Holly Claudia Ott, Dr. Walter Stechele, and Paul Zuber.

Of course, I would have never succeeded in this work without the support of my colleagues from the system administration group, particularly Wolfgang Kohtz, Stephan Herrmann, Fabian Vogelbruch, and Jürgen Foag. Likewise, I want to thank our non-technical staff members Verena Draga, Gabriele Spöhrle, and Doris Zeller for their help with all kinds of administrative issues.

Last but not least, I would like to express special thanks to Wolfgang Höld and Martin Embacher of NATIONAL SEMICONDUCTOR GMBH, Fürstenfeldbruck, Germany, as well as Dr. Bijoy Chatterjee of NATIONAL SEMICONDUCTOR CORPORATION, Santa Clara, USA, for the excellent cooperation. The company's financial support for parts of this work is gratefully acknowledged.

Torsten Mahnke
Munich, May 2003

Abstract

Dual supply voltage scaling (DSVS) for logic-level dynamic power optimization has increasingly attracted attention over the last few years. The first major objective of this work is to demonstrate that DSVS, in contrast to various supply and threshold voltage regulation and multiple threshold voltage techniques, is well suited to the design of standard-cell-based digital CMOS ASICs for the following reasons. Firstly, the technique can be fully automated in the logic synthesis process, which minimizes the additional design time. Secondly, DSVS can easily be integrated with existing ASIC design flows, provided that a suitably modeled dual supply voltage (DSV) standard cell library exists. Thirdly, no specific constraints are imposed on the choice of fabrication process, so that the technique can be applied to any circuit designed for mainstream bulk CMOS technologies.

A novel power-driven logic synthesis methodology is proposed. The idea behind this methodology is to provide a DSV standard cell library that is modeled in such a way that the gate sizing functionality of existing tools can be exploited for DSVS. Since this approach renders dedicated DSVS algorithms superfluous, only little modification of established design flows is required. The important aspects of DSV library development are discussed. This includes the design of level-converting standard cells, different ways of modeling the dynamic power consumption, the modeling of pin connectivity constraints, and the characterization of low voltage and level-converting cells.

The second major objective of this work is to investigate the potential and the limitations of DSVS in a realistic design environment. For this purpose, based on an in-depth analysis of power optimization techniques in general and voltage scaling techniques in particular, a number of state-of-the-art techniques and strategies to be considered in the evaluation of the proposed methodology are identified. Most importantly, DSVS and other logic-level techniques that directly compete with DSVS are used simultaneously in this work.

The fundamental characteristics of DSVS are investigated using 48 combinational and sequential benchmark circuits as test cases. In numerous experiments under strict and moderately relaxed timing constraints, the power reduction due to DSVS is up to 20%. However, the effectiveness of DSVS depends largely on the circuit, and an average power reduction of less than 10% is achieved. A direct comparison with the well-known clustered voltage scaling (CVS) algorithm reveals a greater effectiveness of the proposed methodology. A thorough analysis of the optimization potential of the test cases provides strong evidence of the realistic design environment being the actual reason for the modesty of the power savings observed in this work. In the analysis of the optimization potential, a novel power savings estimation method (PSEM) is used. The PSEM is based on a timing and power analysis theory developed in this work. Other investigated aspects are the impact of the timing constraint strictness on the optimization potential and a comparison with straightforward global supply voltage scaling (GSVS) strategies.

The proposed DSV logic synthesis methodology supports clock voltage scaling. While this technique significantly reduces the dynamic power consumption in the clock network, it

often creates power overheads in other parts of the design. Large overall power savings can be achieved if, on the one hand, the clock network contributes significantly to the total power and, on the other hand, the power overheads are relatively small. Since clock voltage scaling introduces additional delay into all paths, some performance degradation must be expected in the case of designs that are subject to the strictest timing constraints.

The methodology is also used on NATIONAL SEMICONDUCTOR'S 16-bit CompactRISC processor core module. The results of these experiments prove that DSVS can coexist with all common design techniques, including clock gating and the widely-used scan test method, in an industrial design environment. Moreover, the extensive use of clock gating, which is typical of modern microprocessors, makes this design an ideal vehicle for an investigation of the interaction of clock gating and clock voltage scaling.

The results of this study imply the following realistic scenario for successful DSV system design. In a complex hierarchical system composed of numerous modules that are subject to different timing constraints, DSV logic synthesis can be applied to those modules that are subject to strict or moderately relaxed constraints and exhibit optimization potentials large enough to compensate for the overhead caused by the more complex DSV layout. Clock voltage scaling can be applied to a subset of these modules after a careful evaluation of the power savings, the power overheads, and the possible performance degradation. Modules that are sufficiently relaxed to be operated entirely at the lower voltage at the cost of a moderate area overhead should be synthesized and optimized for global low voltage operation in the traditional way. Finally, all the remaining modules must be designed for global high voltage operation.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Application Specific Integrated Circuits and Systems

The integrated circuit (IC) market is divided into the standard product (SP) segment and the application specific integrated circuit (ASIC) segment. Standard products are produced in large volumes and can be used in a variety of applications. Major advantages of using existing SPs are relatively low cost, off-the-shelf availability, and proven reliability. On the other hand, SPs are not optimized for any specific application and are, thus, usually inefficient regarding the performance, the power consumption, and the area. Application specific integrated circuits are optimized for a specific application or application domain. Therefore, ASICs provide performance-, power- and area-efficient implementations of specific functionalities. The efficiency, however, comes at the cost of longer design times, a larger potential for system failure, and a higher price per unit. About 98% of all ASICs are digital circuits fabricated in mainstream complementary metal oxide semiconductor (CMOS) technologies [45].

Application specific integrated circuits are designed using full-custom or semi-custom design styles. In full-custom design, every single transistor is optimized individually. This is an expensive design style seldom used for designing digital ASICs; full-custom ASICs accounted for only 19% of the total ASIC market in 2002 (see Figure 1.1a). The distinct characteristic of semi-custom design is the use of pre-designed or pre-fabricated building blocks. The important semi-custom ASIC types are standard-cell-based ICs, gate arrays, and programmable logic devices (PLD). Most digital ASICs rely on the standard cell concept (54% in 2002; expected to reach 61% by 2007; see Figure 1.1a). In the standard-cell-based design style a variety of pre-designed logic gates, flip-flops, latches, half and full adders, etc. are available in so-called standard cell libraries. The major advantages of the standard cell concept over pre-fabricated gate arrays and PLDs are a higher integration density and a better performance. Compared with full-custom design, the standard cell concept raises the level of abstraction, which significantly reduces the design times and

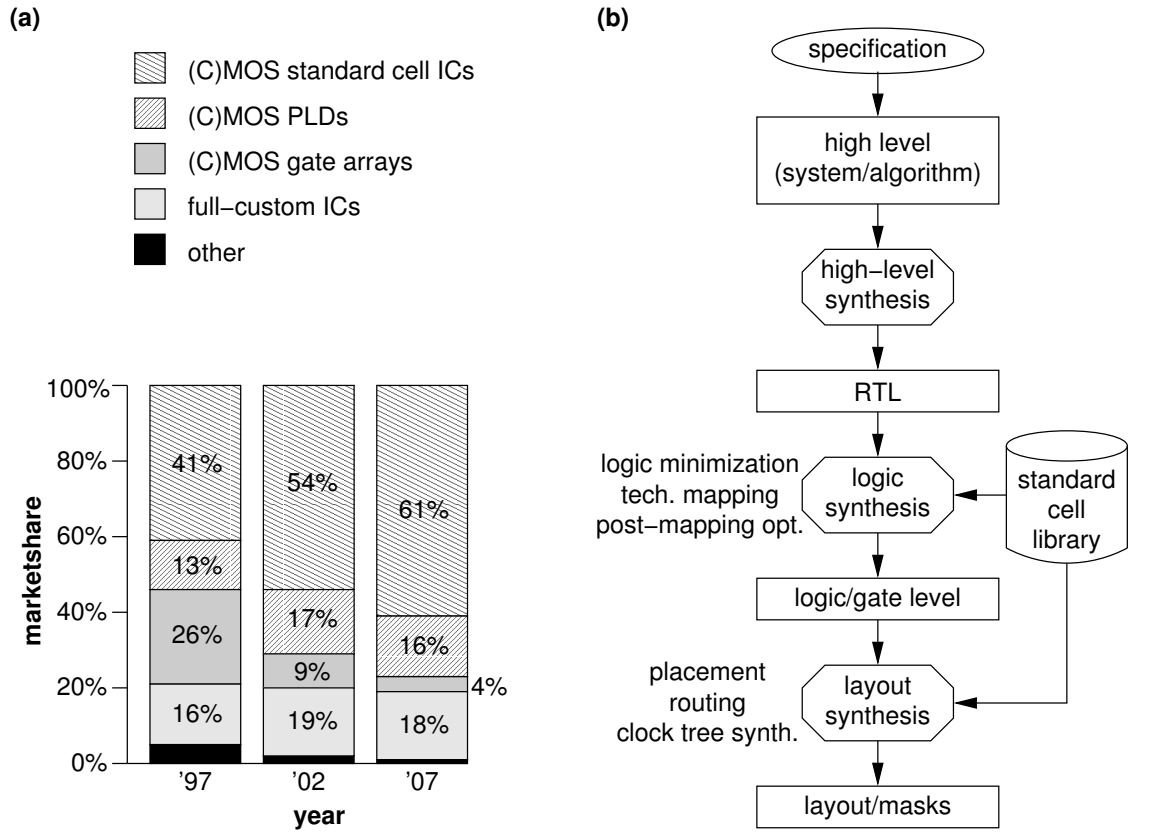

Figure 1.1: Application specific integrated circuits (ASIC): (a) market share of different ASIC types [46]; (b) standard-cell-based ASIC design flow [32, 37].

facilitates the development of tools for automated design and optimization. In recent years, robust and largely automated methodologies for the design of standard-cell-based ASICs have been established, which is an important reason for the increasing use of ASICs in all kinds of electronic applications [37].

A typical standard-cell-based ASIC design flow is depicted in Figure 1.1b. The different levels of the design hierarchy are clearly visible. The design follows a top-down approach and starts with a specification of the required functionality and constraints. The specification is then translated into a high-level description written in a programming or hardware description language (HDL). This high-level model serves two purposes. Firstly, the specification can be verified and different algorithms providing the same function can be evaluated. Secondly, the model can be used as a reference for the verification of lower level design descriptions in subsequent phases of the design process.

High-level synthesis is the task of generating a design description at the register transfer level (RTL) from the high-level model. Although both academic and commercial tools exist, automated high-level synthesis has not yet become an integral part of industrial design

flows. Thus, RTL modeling using HDLs such as VHDL (Very High Speed Integrated Circuit Hardware Description Language), Verilog, or SystemC still has to be done manually.

The remaining design steps are largely automated. The RTL model is first mapped to a logic-/gate-level implementation using a logic synthesis tool and elements from a standard cell library. The result of this logic synthesis step is a technology dependent gate-level netlist. In the subsequent layout synthesis, the cells that appear in the netlist are (virtually) placed on the surface of the die, and the interconnections between the cells are routed. Interconnect routing means that the exact shapes and locations of all wires are defined. The final result of the design process is a set of data providing all the information required for fabricating the production masks.

The design flow described above is actually not restricted to ASIC design. In fact, as the quality of the design tools improves and as the maximum complexity of standard-cell-based designs increases, thus reducing the quality gap between full-custom and semi-custom designs, the standard cell concept is more and more adapted to the design of most digital parts of modern systems-on-a-chip (SoC), including embedded microprocessor, microcontroller and digital signal processor (DSP) cores. Even SPs are sometimes designed using standard cells instead of full-custom design styles in order to reduce the design time.

1.2 Importance of Low Power Design

There has been tremendous progress in semiconductor technology since the first ICs were introduced in the 1960's. The minimum feature size, i.e. the minimum dimension of the semiconductor structures, has become smaller and the die sizes have increased. Consequences of this technology scaling trend are reduced device capacitances, higher integration densities, performance improvements, and increased circuit complexities.

In the past, the circuit performance and the chip area were the major issues in IC design. This has changed over the last ten years. The power consumption is now another major design criterion. This development has been driven mainly by the rapid growth of the portable consumer electronics market, where system running time, battery weight, and battery volume are critical parameters. The aforementioned increase in integration density and circuit performance, however, has led to enormous on-chip power and power densities, as indicated by the two graphs in Figure 1.2[1]. Since excessive total power and power density cause serious reliability problems, the power consumption is no longer a specific problem of mobile applications. In fact, it is equally critical, if not more, in the design of high-performance ICs for non-battery-powered applications.

[1]The primary sources for the data included in the figure are F. Pollack's Micro32 keynote speech [92], the Microprocessor Quick Reference Guide (http://www.intel.com/pressroom/kits/quickreffam.htm), the Processor Spec Finder (http://processorfinder.intel.com), and various data sheets (http://www.intel.com).

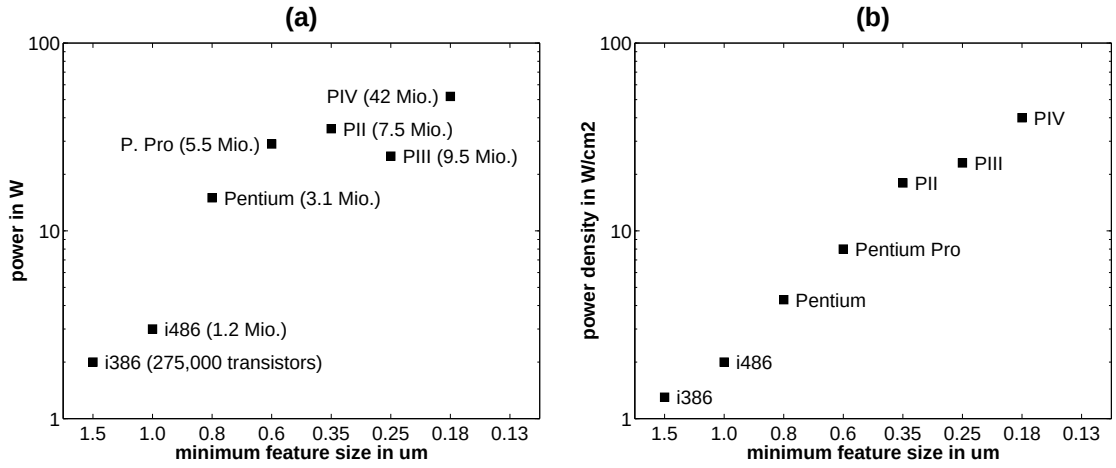

Figure 1.2: History of the power consumption and the complexity of INTEL'S desktop processors: (a) absolute power and number of transistors per chip at the date of introduction; (b) power density at the date of introduction.

The evolution of the battery technology is slow compared with the progress in semiconductor technology, so that the developers of portable applications cannot count on significant improvements in this field as a solution to the running time issue. The power consumption of existing portable or non-portable applications can be reduced by transferring selected ICs to a next-generation fabrication technology. Provided that the functionality and the performance of the ICs remain unchanged, the technology-scaled implementations dissipate less power because of smaller circuit capacitances and lower supply voltages. On the other hand, when technology scaling is exploited for maximizing the circuit complexity and the performance in order to meet the requirements of new applications, the power consumption actually increases, as discussed in the preceding paragraph. Ceramic packages, cooling fans, or other sophisticated cooling means may be used for removing the heat from such complex high performance ICs, so as to guarantee reliability. Expensive external cooling means, however, increase the overall system cost and should, therefore, be avoided.

It follows from the above arguments that neither improvements in battery or semiconductor technology nor external cooling solve the fundamental problem of power consumption in the development of portable or non-portable applications. Therefore, low power design methods play an important role in IC design.

1.3 Scope and Objective of this Work

In many applications, the dynamic power consumption is dominant. The embedded microcontroller system used for a case study in this work is an example of such an application.

The dynamic power consumption depends quadratically upon the supply voltage. Thus, supply voltage scaling is generally considered an effective means of power optimization. In existing industrial ASIC design flows, however, voltage scaling is usually limited to straight-forward global supply voltage scaling approaches.

The first objective of this work is to develop a new methodology of optimizing the dynamic power consumption of standard-cell-based ASICs by means of an advanced voltage scaling approach. The methodology to be developed should fulfill three requirements:

1. The power optimization should be fully automated in order to minimize the additional design time. Since automatic synthesis in industrial standard-cell-based design flows is usually restricted to logic and layout synthesis, this first requirement implies that a logic-level voltage scaling approach should be implemented.

2. Standard-cell-based design at the logic level is quasi standardized and is usually based on commercial tools. Thus, a second requirement is to rely on the existing tools as far as possible, so as to facilitate the integration of the novel methodology with existing industrial design flows.

3. Constraints preventing standard bulk CMOS fabrication processes from being used are not allowed to be introduced.

Dual supply voltage scaling (DSVS) at the logic level is a suitable and promising candidate for the development of a methodology that meets the above requirements. Logic-level DSVS can be split into two separate tasks: dual supply voltage (DSV) logic synthesis and DSV layout synthesis. This work aims at the development of a methodology for DSV logic synthesis.

The second objective of this work is to investigate the potential and the limitations of DSVS in a realistic design environment. Particularly, the true additional benefit of DSVS in comparison with *state-of-the-art* power-driven logic synthesis is to be investigated. At this point, the term *state-of-the-art* shall be defined in order to avoid ambiguity. The term is used throughout this work to characterize methods and tools that are commonly used in existing industrial standard-cell-based design flows.

1.4 Outline

In **Chapter 2**, the fundamentals of digital CMOS circuit behavior, including the gate delay and the sources of power consumption, are explained. This knowledge is needed for understanding the concepts presented in the remainder of this document. The theory is based on an alpha-power-law transistor model. A consistent set of equations is obtained by introduction of fitting parameters that guarantee transregional continuity of the drain current

equations and by derivation of a novel expression describing the short-circuit power. On the basis of this theory, basic low power design strategies are discussed.

Chapters 3 and 4 are devoted to a discussion of low power design techniques in general and voltage scaling techniques in particular, so as to provide an overview of the broad field of low power design and to point out the state of the art in the standard-cell-based design of low power ASICs. This discussion also motivates a reasonable choice of power optimization techniques to be considered in the evaluation of the methodology proposed in this work. The selected set of power optimization methods includes logic-level DSVS, global supply voltage scaling (driven by logic-level parallelization and gate up-sizing), clock gating, and various technology-dependent logic-level techniques.

In **Chapter 5**, the fundamentals of DSV logic synthesis are explained. Timing conditions for the applicability of voltage scaling to individual gates at the logic level are formulated, and an expression describing the expected power savings is derived. The extension of DSV logic synthesis to voltage scaling in the clock network is explained as well. This includes the development of an expression describing the power savings and the power overheads associated with clock voltage scaling. The discussion of DSV logic synthesis and clock voltage scaling is followed by a review of relevant related work. Finally, existing solutions to the DSV layout synthesis problem are discussed in order to emphasize the general feasibility of the entire concept of DSVS at the logic level.

In **Chapter 6**, a novel DSV logic synthesis methodology and guidelines for the development of DSV standard cell libraries are presented. The idea behind this methodology is to facilitate the integration of DSVS with state-of-the-art power-driven logic synthesis. Thus, the experiments carried out using the novel methodology reveal the true additional benefit of DSVS, which is crucial for a realistic evaluation of the potential and the limitations of DSV logic synthesis.

The library development guidelines cover different ways of modeling the dynamic power consumption, the modeling of pin connectivity constraints, and the characterization of low voltage and level-converting cells. Furthermore, the circuit structure as well as the timing and power characteristics of level-converting inverters, buffers, and flip-flops, including a novel level-converting scan-flip-flop with clear and preset inputs, are discussed. The cells exploit a new pull-up circuit technique that improves the timing characteristics.

A power savings estimation method, which has been developed for analyzing the optimization potential of the proposed methodology, is also presented in this chapter.

In **Chapters 7 and 8**, the results of experiments carried out using the proposed methodology are discussed. A variety of combinational and sequential benchmark circuits and a real embedded microcontroller system serve as test cases in the experiments. The results are carefully compared with the results of related work. The potentials and the limitations of DSV logic synthesis including clock voltage scaling are thoroughly analyzed under consideration of various state-of-the-art design and optimization techniques.

Chapter 9 provides conclusions and an outlook on future work.

Chapter 2

Transistor Current, Gate Delay, and Power Consumption

The power consumption and the performance of digital CMOS circuits are determined by the drain currents of the transistors. Therefore, an empirical model describing the current in different regions of operation is presented first. A gate delay formula that was derived from the said drain current model is presented next. Finally, the sources of power consumption and basic optimization strategies are discussed.

2.1 Drain Current

Digital CMOS circuits are composed of n-channel and p-channel enhancement-type metal oxide semiconductor field effect transistors (MOSFET). Figure 2.1 shows the respective schematic symbols and terminal names. In the following paragraphs, the basics of the electrical behavior of n-channel transistors are described. The same equations are valid for p-channel transistors, if absolute values are used for all voltages and the direction of the drain current is reversed.

Depending on whether the gate-source voltage V_{GS} is smaller or larger than the threshold voltage V_t, the n-channel transistor is said to be in the below- or the above-threshold regime, respectively. The drain current in the below-threshold (subthreshold) regime is known to have an exponential characteristic [5, 13, 17, 89]. In the above-threshold regime, the linear and the saturation region are distinguished. The drain current in these two regions is given by the widely-used alpha-power-law model [13, 95].

Subthreshold regime. The drain current I_D of n-channel transistors in the subthreshold regime (subthreshold current I_{DSUB}) can be written as

$$I_{DSUB} = I_{DSUB0} \cdot e^{(V_{GS}-V_t)/(nV_{TH})} \cdot \left[1 - e^{-V_{DS}/V_{TH}}\right] \tag{2.1}$$

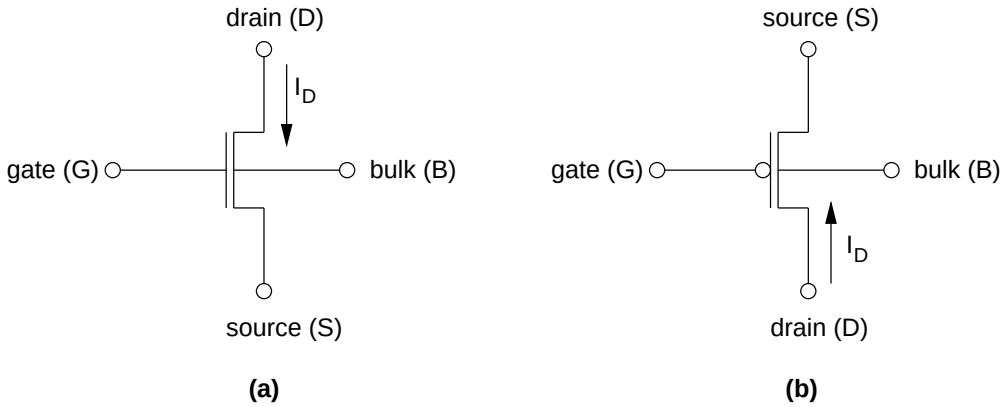

Figure 2.1: Symbols and terminal names for enhancement-type MOSFETs: (a) n-channel MOSFET; (b) p-channel MOSFET.

with

$$I_{DSUB0} = (W/L)\,\mu\,C_{ox}\,n\,V_{TH}^2\,K_{SUB} \quad , \tag{2.2}$$

$$C_{ox} = \varepsilon_{ox}/t_{ox} \quad , \tag{2.3}$$

and

$$n = 1 + \varepsilon_{si}/(C_{ox}\,D) \quad . \tag{2.4}$$

In Equation 2.1, I_{DSUB0} is the drain current at V_{GS} equal to V_t, V_{DS} is the drain-source voltage, V_{TH} is the thermal voltage (26 mV at a temperature of 300 K), and n is a process parameter. In Equation 2.2, W and L are the gate width and length respectively, μ is the electron mobility, C_{ox} is the gate oxide capacitance per unit area, and K_{SUB} is a unitless fitting parameter that can be adjusted for continuity between the below- and above-threshold regimes. In Equation 2.3, ε_{ox} and t_{ox} are the dielectric constant and the thickness of the gate oxide, respectively. Finally, in Equation 2.4, ε_{si} is the dielectric constant of the silicon and D is the thickness of the depletion layer in the silicon under the gate electrode.

The subthreshold current is proportional to the gate width-to-length ratio and depends exponentially on the gate-source and threshold voltages. The influence of the drain-source voltage can be neglected in most practical cases, because the expression $1 - e^{-V_{DS}/V_{TH}}$ approximates to 1, even for relatively small V_{DS} of about 100 mV for instance.

Saturation region. The term alpha-power-law model stems from the following representation of the drain current in the saturation region of the above-threshold regime [13, 95]:

$$I_D = \beta\,K_{ISAT}\,(V_{GS} - V_t)^{\alpha} \tag{2.5}$$

In Equation 2.5, K_{ISAT} is a fitting parameter (unit: $\mathrm{V}^{2-\alpha}$). The velocity saturation index α is a process parameter that can take on values between one (all carriers move at saturation

velocity) and two (no velocity saturation). The transconductance parameter β depends on the gate width-to-length ratio, the electron mobility, and the gate oxide capacitance per unit area, as in

$$\beta = (W/L)\,\mu\,C_{ox} \quad . \tag{2.6}$$

A transistor is in the saturation region if the drain-source voltage V_{DS} is larger than or equal to the saturation voltage V_{DSSAT}. The latter is defined as

$$V_{DSSAT} = K_{VSAT}\,(V_{GS} - V_t)^{\alpha/2} \quad , \tag{2.7}$$

where K_{VSAT} is a fitting parameter (unit: $V^{1-\alpha/2}$). For transregional continuity, K_{VSAT} must be chosen such that

$$K_{VSAT} = K_{ISAT}/K_{ILIN} \quad , \tag{2.8}$$

where K_{ILIN} is another fitting parameter defined in the following paragraph.

Linear region. In the alpha-power-law model [13, 95], the drain current of n-channel transistors in the linear region of the above-threshold regime is expressed as

$$I_D = \beta\,K_{ILIN}\,V_{DS}\,(V_{GS} - V_t)^{\alpha/2} \quad , \tag{2.9}$$

where K_{ILIN} is a fitting parameter (unit: $V^{1-\alpha/2}$).

A transistor is in the linear region if the drain-source voltage V_{DS} is smaller than the saturation voltage V_{DSSAT} defined in the previous paragraph.

Threshold voltage and body effect. The threshold voltage of n-channel transistors is given by

$$V_t = V_{t0} + \gamma\left[\sqrt{|2\Phi_F + V_{SB}|} - \sqrt{|2\Phi_F|}\right] \quad , \tag{2.10}$$

with

$$V_{t0} = V_{FB} + 2\Phi_F + \gamma\sqrt{|2\Phi_F|} \quad , \tag{2.11}$$

$$\Phi_F = V_{TH}\,\ln\frac{N_A}{n_i} \quad , \tag{2.12}$$

and

$$\gamma = \left(\sqrt{2qN_A\varepsilon_{si}}\right)/C_{ox} \quad . \tag{2.13}$$

In Equation 2.10, V_{t0} is the threshold voltage for zero source-bulk voltage V_{SB}, γ is the substrate or body factor, and Φ_F is the difference between the Fermi level and the intrinsic Fermi level of the semiconductor. In Equation 2.11, V_{FB} is the flat band voltage. In Equations 2.12 and 2.13, n_i is the intrinsic carrier density, N_A is the doping concentration in the p-type substrate, and q is the elementary charge.

The same expressions can be used for describing the threshold voltage of p-channel transistors, if the donator concentration N_D is substituted for the acceptor concentration N_A and the signs of Φ_F and γ are reversed [43, 110].

The threshold voltage increases with increasing source-bulk voltage, which leads to reduced drain current. This behavior is known as the substrate or body effect. It may have large impact on gate delays and static power consumption.

2.2 Gate Delay

The input-to-output delay t_D of an inverter driving a load capacitance C_{node} at a supply voltage V_{DD} can be derived using Equation 2.5. The solution consists of an input transition time (t_T) dependent term and an output load capacitance (C_{node}) dependent term and is given by

$$t_D = \left(\frac{1}{2} - \frac{1 - V_t/V_{DD}}{\alpha + 1} \right) t_T + \frac{C_{node} V_{DD}}{2\beta K_{ISAT} (V_{DD} - V_t)^\alpha} \quad . \tag{2.14}$$

A detailed derivation of Equation 2.14 explaining all assumptions and simplifications can be found in [95] and in Appendix A.1.

Assuming an ideal step function at the input, i.e. t_T is equal to zero, the expression for t_D can be simplified to

$$t_D = \frac{C_{node} V_{DD}}{2\beta K_{ISAT} (V_{DD} - V_t)^\alpha} \quad . \tag{2.15}$$

According to Equation 2.15, the delay increases with increasing load capacitance. Supply voltage reduction also leads to larger delay. On the other hand, larger gate width-to-length ratios and smaller threshold voltages reduce the delay. Note that the body effect described above results in a larger threshold voltage and, hence, a larger delay.

Equation 2.15 is often used as an approximation of the delay of any type of CMOS gate. Thus, β can be seen as an effective transconductance parameter representing the current source and sink capability of the symmetrical pull-up and pull-down networks of the gate.

2.3 Power Consumption

The total power consumption P_{tot} of digital CMOS circuits can be written as the sum of the dynamic and static components P_{dyn} and P_{stat}:

$$P_{tot} = P_{dyn} + P_{stat} \tag{2.16}$$

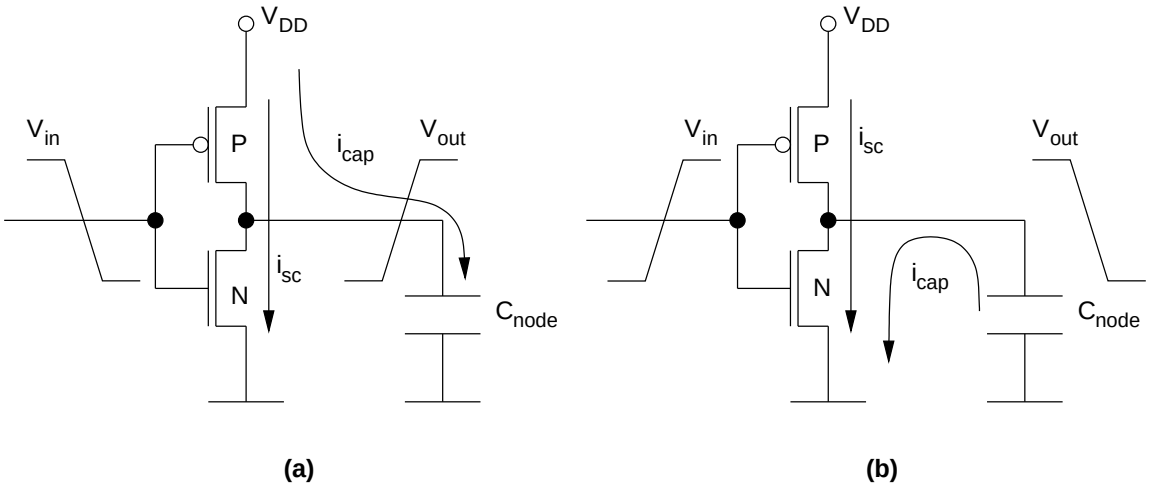

Figure 2.2: Sources of dynamic power consumption: (a) short-circuit current and current charging a node capacitance; (b) short-circuit current and current discharging a node capacitance.

In the majority of applications the dynamic power consumption is dominant. It can be further split into the capacitive component P_{cap} and the short-circuit component P_{sc}:

$$P_{dyn} = P_{cap} + P_{sc} \tag{2.17}$$

The short-circuit power has long been considered contributing less than 10% to the total dynamic power consumption [17]. In modern standard-cell-based ASICs the contribution of the short-circuit power may be larger but the capacitive power remains dominant [121].

2.3.1 Capacitive Switching Power

The capacitive component of P_{dyn} is due to currents that charge or discharge capacitances associated with circuit nodes, as illustrated in Figure 2.2. During the falling transition at the input node of the inverter, the current i_{cap} flows from the power supply through the p-channel transistor to the capacitor at the output node. The capacitor is charged and the output voltage rises to V_{DD}. The total energy drawn from the power supply is $C_{node} \cdot V_{DD}^2$, where C_{node} is the total capacitance associated with the output node of the inverter [17][1]. One half of the energy is dissipated in the p-channel transistor and the other half is stored in the capacitor. During the rising transition at the input, the current i_{cap} flows from the capacitor to ground and the energy previously stored in the capacitor is dissipated in the n-channel transistor. No additional energy is drawn from the supply.

[1]The derivation is reproduced in Appendix A.2.

The total energy drawn from the supply during a period of m clock cycles is the product of the number of occurrences of zero-to-one output transitions $k(m)$ and the energy drawn during one such transition $C_{node} \cdot V_{DD}^2$. The average capacitive switching power dissipated during the said period of time is, thus, given by

$$P_{cap} = \frac{k(m)\, C_{node}\, V_{DD}^2}{m\, T_{clk}} \quad , \tag{2.18}$$

where T_{clk} is the clock period. The quotient $k(m)/m$ is the average number of occurrences of zero-to-one transitions per clock cycle. This is often interpreted as the probability of a zero-to-one transition occurring in a clock cycle. Introducing a switching activity factor α_{01} that denotes the said probability, Equation 2.18 can be rewritten as

$$P_{cap} = \alpha_{01}\, f_{clk}\, C_{node}\, V_{DD}^2 \quad , \tag{2.19}$$

where $f_{clk} = 1/T_{clk}$ is the clock frequency. Equation 2.19 was derived considering only one circuit node. An equivalent expression, however, can be formulated for any node in a complex circuit, and the total capacitive switching power can be calculated by summing Equation 2.19 over all nodes:

$$P_{cap} = f_{clk}\, V_{DD}^2 \sum_{i=1}^{N} \alpha_{01,i}\, C_{node,i} \tag{2.20}$$

In Equation 2.20, N is the total number of nodes in the circuit, and $\alpha_{01,i}$ and $C_{node,i}$ are the switching activity and the capacitance associated with the i-th node, respectively.

As illustrated in Figure 2.3, nodes in synchronous circuits make one transition per clock cycle at the most if no spurious transitions occur. A zero-to-one transition may occur every second clock cycle at the most. Thus, a maximum of 0.5 can be expected for the switching activities $\alpha_{01,i}$. The actual values depend on the input pattern (stimuli) and the circuit topology [17]. In practice, the switching activities are often calculated as $k_i(m)/m$ where the number of occurrences of zero-to-one transitions at the i-th node $k_i(m)$ is obtained from simulations using typical input pattern.

So far, only useful transitions were considered. However, if the signals at different input pins of a logic gate have different arrival times, spurious transitions (so-called ”glitches”) may occur at the output of the gate. These transitions are not useful regarding the functionality of the circuit. Nevertheless, the transitions contribute to the capacitive switching power. Because of glitching, circuit nodes may switch several times in one clock cycle before settling, which results in switching activities larger than 0.5.

For the purpose of power analysis in digital circuits, a node capacitance C_{node} is usually represented by three lumped capacitances connected between the respective node and ground. These are the total gate-to-channel capacitance C_G of the transistors that have their gate directly connected to the node, the total drain-to-bulk diffusion capacitance C_{DB} of the

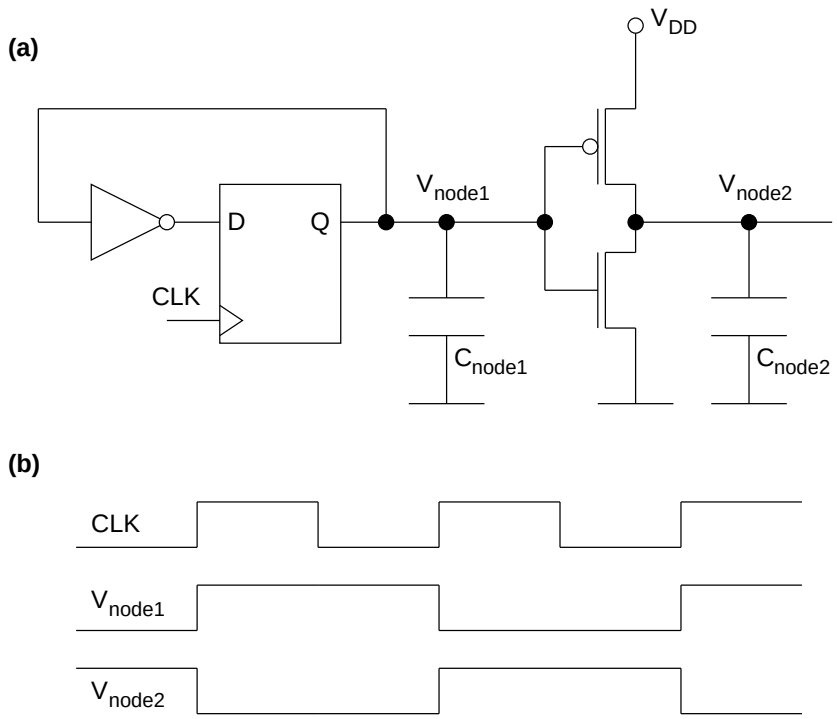

Figure 2.3: Definition of the switching activity in synchronous circuits: (a) toggling D-flip-flop; (b) clock (CLK) and node voltage ($V_{node1/2}$) waveforms.

transistors that have their drain directly connected to the node, and the interconnect capacitance C_{int} that includes capacitances between the wire and the layers above and below as well as coupling capacitances to neighboring wires:

$$C_{node} = C_G + C_{DB} + C_{int} \tag{2.21}$$

2.3.2 Short-Circuit Power

In this section, a novel expression describing the short-circuit power is presented. The expression was derived[2] on the basis of the alpha-power-law MOSFET model, so as to be consistent with the set of equations presented in this chapter.

Ideally, n-channel and p-channel transistors never conduct simultaneously in digital CMOS circuits. This requires ideal step waveforms at the gate electrodes of all transistors. In reality, however, the signals exhibit finite rise and fall times, as shown in Figures 2.2 and 2.4. As a result, n-channel and p-channel transistors are simultaneously on for a short period

[2]The detailed derivation can be found in Appendix A.3.

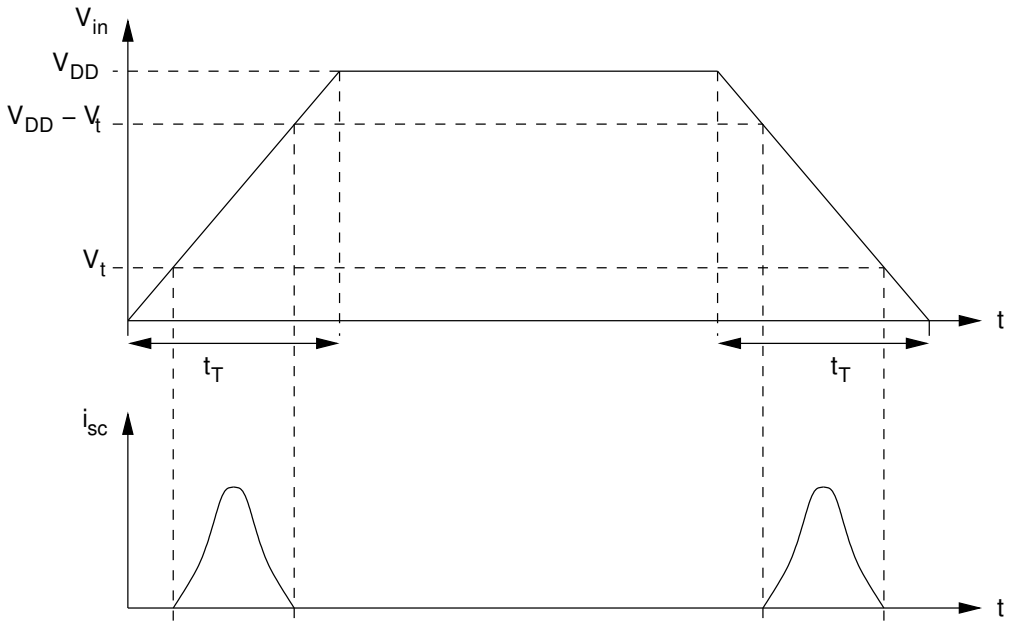

Figure 2.4: Short-circuit current for an inverter without output load capacitance [17].

of time during switching, i.e. while V_{in} is greater than V_t and smaller than $V_{DD} - V_t$. This establishes a direct path from the power supply to ground.

The waveform of the short-circuit current i_{sc} is depicted in Figure 2.4. In the case of zero output load capacitance, the short-circuit power consumption of a circuit composed of N gates is given by

$$P_{sc} = \sum_{i=1}^{N} \alpha_{01,i} \, f_{clk} \, \frac{4 \beta_i K_{ISAT,i}}{(\alpha+1) \, 2^{\alpha+1}} \, t_{T,i} \, (V_{DD} - 2 V_t)^{\alpha+1} \quad . \tag{2.22}$$

According to Equation 2.22, the short-circuit power consumption is a function of the switching activity, the clock frequency, the effective transconductance and, hence, the dimensions of the transistors, the input signal transition time, the supply voltage, and the threshold voltage. Unfortunately, the analytical analysis of the short-circuit power taking into account an output load capacitance does not lead to a similarly compact solution.

The impact of a non-zero load capacitance C_{node} can be explained with the help of Figure 2.5. In the case of a falling edge at the input, the short-circuit current is determined by the gate-source and drain-source voltages at the n-channel transistor.

A large load capacitance C_{node} causes a slow transition of the output voltage V_{out}. If the output transition time is much larger than the input transition time, as shown in Figure 2.5a, the drain-source voltage at the n-channel transistor V_{DSN} is small, and the n-channel transistor is in the linear region most of the time while both transistors are conducting; the result is a relatively small short-circuit current.

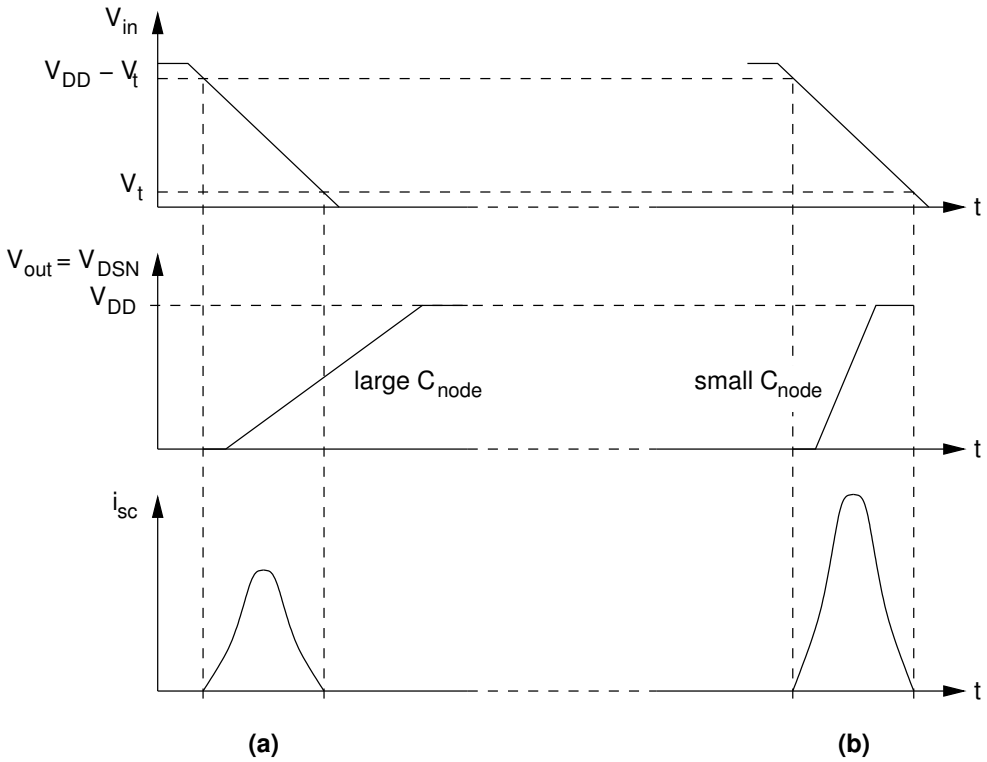

Figure 2.5: Impact of the output signal slope on the short-circuit current: (a) slow output transition caused by large output load; (b) fast output transition caused by small load.

A small load capacitance causes a fast transition of the output voltage. If the output transition time is much shorter than the input transition time, as shown in Figure 2.5b, V_{DSN} rises quickly towards large values, the n-channel transistor saturates while both transistors are still conducting, and the resulting short-circuit current becomes relatively large. Results of Spice simulations that confirm the above arguments can be found in [121].

2.3.3 Static Power

The static power consumption P_{stat} is mainly due to the subthreshold currents discussed in Section 2.1. It can be expressed as

$$P_{stat} = V_{DD} \cdot I_{DSUB} \tag{2.23}$$

$$= V_{DD} \cdot I_{DSUB0} \cdot e^{(V_{GS}-V_t)/(nV_{TH})} \quad . \tag{2.24}$$

As mentioned before, the subthreshold currents depend exponentially on the gate-source and threshold voltages. To be more exact, for a given V_{GS}, I_{DSUB} increases exponentially with decreasing V_t. This must be taken into account when reducing the threshold voltage

for performance reasons. Note that, because of the exponential characteristic of I_{DSUB}, the body effect discussed in Section 2.1 has a large impact on the static power consumption.

For low leakage currents, it is also important to use transistors with steep transfer characteristics in the subthreshold regime. This is commonly measured by the slope $1/S_t$ of the semi-logarithmic transfer characteristic, i.e. $\log(I_{DSUB})$ versus V_{GS}, which is linear in the subthreshold regime. The larger the slope, i.e. the smaller S_t, the closer the transistor's behavior is to that of ideal switches. The reciprocal subthreshold slope S_t is approximately [17]

$$S_t = n V_{TH} \ln(10) \quad . \tag{2.25}$$

According to Equation 2.4, n cannot be smaller than one and, hence, S_t has a lower bound given by

$$S_t \geq V_{TH} \ln(10) \quad . \tag{2.26}$$

The lower bound is 60 mV[3] at room temperature (300 K). Bulk CMOS technologies typically have reciprocal subthreshold slopes of up to 100 mV [17, 31], whereas this parameter can be close to the minimum in silicon on insulator (SOI) technologies [31].

The static power due to reverse-biased diode leakage is usually negligible [121].

2.4 Basic Low Power Design Strategies

The choice of supply voltage affects all components of power. It has, however, the largest impact on the dynamic power consumption because of the non-linear dependence of P_{dyn} and V_{DD} described by Equations 2.19 and 2.22. Therefore, supply voltage scaling is considered a very effective means of dynamic power reduction.

The capacitive component of power P_{cap} can also be optimized by reduction of the effective switched capacitance $\alpha_{01} \cdot C_{node}$ or by clock frequency (f_{clk}) scaling. The latter can often be combined with supply voltage scaling for an even larger power reduction.

The short-circuit power P_{sc}, just as the capacitive power, can be optimized by reduction of the switching activity α_{01} or by reduction of the clock frequency f_{clk}. Another important parameter is the input transition time t_T. Finally, smaller transistors, i.e. transistors with smaller channel width W, result in reduced P_{sc}.

Although the threshold voltage V_t has an impact on the short-circuit power P_{sc}, circuit performance and static power considerations have priority over P_{sc} regarding the choice of V_t. A slightly larger V_t results in significantly less static power P_{stat} at the cost of larger delays t_D. This is obvious from Equations 2.1 and 2.15. The sizes of the transistors also affect P_{stat}. The most important principle of static power optimization, however, is to separate inactive circuits from the power supply so that no current can flow at all.

[3] This is often written as 60 mV/decade, in order to express that the gate-source voltage needs to be changed by 60 mV to change the drain current by one decade.

Chapter 3

Low Power ASIC Design

3.1 Overview

The development of low power integrated systems requires several fundamental design decisions to be taken and a combination of different power optimization techniques to be applied to the system or to parts thereof.

Throughout the last ten years, numerous approaches to low power design have been proposed. These include software as well as hardware optimization strategies. Regarding hardware optimization, further distinction can be made between techniques that are intended for the design of logic circuits and techniques that are specific to memory. The work underlying this thesis is focused on low power design of logic circuits.

Figure 3.1 provides an overview of low power design techniques frequently discussed in the literature[1]. It is evident from the figure that, in a hierarchical design flow such as the one introduced in Section 1.1, power reduction can be achieved at all levels of abstraction. Although high-level power optimization is believed to be most effective, the improvements that can be achieved at the lower levels are none the less significant [59]. Thus, any low power design methodology should include a set of high- and low-level optimization techniques that complement one another.

Very few low power design techniques have been established as standard (state-of-the-art) techniques in the development of real applications. Others have proven to be feasible in experimental designs. Many techniques, however, still are of purely academic importance.

For the implementation and evaluation of a new optimization technique, it is important to identify those state-of-the-art techniques (at the same or at a different level of abstraction) that may come into conflict with the new method or may have an impact on the effectiveness of the novel technique.

[1] References are given in the remainder of this chapter.

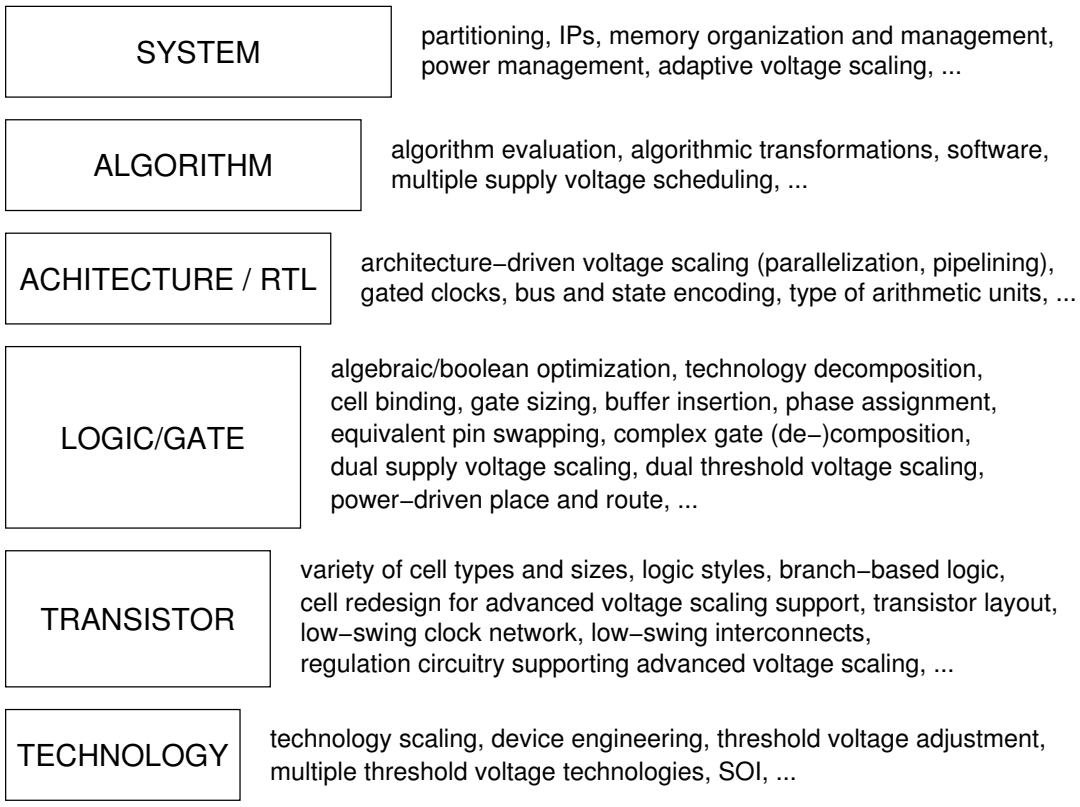

Figure 3.1: Low power design techniques.

In this chapter, low power design methods are reviewed, their suitability for ASIC design is discussed, and state-of-the-art techniques to be considered in the evaluation of the proposed methodology are identified. Accommodating the focus of this study, the survey of low power design methods is continued in Chapter 4 with a separate discussion of voltage scaling techniques.

3.2 Fundamental Design Decisions

The development of electronic systems usually starts with the specification. At this early stage in the design process, all the information required for developing a working product that fits into a specific market segment is gathered. This includes the functionality, the performance, and the type of power supply.

From this information, conclusions regarding the power consumption can be drawn and appropriate constraints can be derived. For instance, if a battery was chosen for power supply, the power consumption must be minimized in order to allow for a reasonable system running time. In the case of very high performance ICs, the power consumption must also be

constrained in order to prevent thermal failure. Clearly, the more demanding the specification, the more design and optimization effort is required for meeting the power constraints. Therefore, the specification should always strictly reflect the actual requirements of the application.

The fabrication technology also has to be chosen at this stage, i.e. before entering the actual design process. A suitable choice can usually be determined on the basis of the specification and any derived constraints.

Mainstream bulk CMOS technologies enable high integration density and high performance at low cost and, at the same time, keep the power consumption at a moderate level. Also, many power optimization techniques can be applied to bulk CMOS designs. For these reasons, bulk CMOS is and will remain to be the technology of choice in the development of most digital electronic systems [58].

Some low power design techniques, however, require enhanced CMOS technologies. For dual threshold voltage scaling, for instance, low and high threshold voltage transistors must be available, as in so-called multiple threshold voltage CMOS technologies.

If the power consumption is extremely critical, silicon on insulator (SOI) technologies can be used instead of bulk CMOS. The expensive SOI wafers and the low yield, however, significantly increase the cost [26].

Once the type of technology has been chosen, the technology level is out of reach for the designer, and all design optimization has to be carried out at the higher levels of abstraction from the transistor level up to the system level. All design and optimization techniques used in this work are compatible with mainstream bulk CMOS fabrication processes.

3.3 System and Algorithmic Level

Partitioning. At the highest levels of abstraction, i.e. the system level and the algorithmic level, the most important task is partitioning. First of all, most systems can be split into logic and memory. The size, the type, the detailed organization, and the management of the memory must then be chosen such that the specified functionality and performance are assured. These choices also have an impact on the power consumption of the system. For further information on low power memory design see the literature [7, 34, 50].

Regarding the logic, which is in the focus of this work, a common approach is to start with functional partitioning, i.e. splitting the specified functionality into less complex subfunctions that can be separately realized by means of different algorithms. The functional partitioning is followed by the actual physical partitioning, where a suitable form of hardware implementation is chosen for each functional partition.

Implementation alternatives. Typical hardware implementation alternatives are general purpose microprocessors, DSPs, application specific microprocessors and microcontrollers, configurable logic, and dedicated hardware modules. Each implementation alternative has its own strengths and weaknesses regarding performance, power consumption, flexibility, time to market, and cost.

A general purpose microprocessor provides maximum flexibility and sufficient performance for many applications. Since such processors are readily available as separately packaged chips for board-level system development or as intellectual property (IP) blocks for SoC design, even the implementation of complex functionalities takes fairly short time. However, the efficiency of general purpose microprocessors in terms of area and power in proportion to performance is usually low.

Digital signal processors (DSP) and application specific processors or controllers are less flexible and, thus, less complex than general purpose processors. If a maximum of flexibility is not absolutely needed, these types of processors lead to more power and area efficient implementations.

Configurable logic is a good choice if time to market is critical, the number of pieces to be fabricated is low and the requirements regarding performance and hardware complexity are moderate. Rapid prototyping is another typical field of application. Unfortunately, hardly any power optimization techniques are applicable to configurable logic.

Maximum performance and minimum power consumption can be achieved only with dedicated hardware. This comes at the expense of increased time to market and cost.

The above statements indicate that the best choice of hardware implementation alternative depends on the specified functionality and performance, the power constraints, and other aspects such as time to market and cost. In modern SoC design, typically some or all components of the system are bought from IP vendors. If the power consumption is critical, it is particularly important to chose IP blocks that have already been designed with the power consumption in mind or that can at least be further optimized, for instance during logic synthesis.

This study is focused on those types of hardware that can be designed and optimized by means of typical ASIC design flows. These are dedicated hardware, application specific processors/controllers and any type of synthesizable IP block (soft macro).

Algorithms and algorithmic optimization. A specific functionality can often be realized through several alternative algorithms. Different algorithms usually exhibit different characteristics regarding the performance, the accuracy, and the power consumption. This should be taken into account in system design. On the other hand, the characteristics of the algorithms are often affected by the choice of hardware implementation alternative and vice versa. Thus, a thorough evaluation of algorithms is a complex and time-consuming task for which standard recipes cannot be formulated and that is, therefore, impossible to

be automated. System designers often bypass the investigation of different combinations of algorithms and hardware implementation alternatives. Instead, previously published research results are adopted, if available and applicable, which usually results in suboptimal solutions.

Once a particular algorithm has been chosen, it can be further optimized with regard to performance or power consumption or both. However, algorithmic optimization techniques are also specific to the type of target hardware. If, for instance, the target is some kind of processor, algorithmic power optimization is a question of software development rather than a hardware design problem. If, on the other hand, the algorithm is to be implemented in dedicated hardware, algorithmic speed-up transformations or multiple supply voltage scheduling can be applied in order to minimize the dynamic power consumption. More details on these techniques can be found in Chapter 4.

Power management. Power management reduces the amount of energy wasted whenever parts of a system are not needed at all or not at full speed. With power management schemes the functionality and the performance of a system or circuit are adjusted to time-variant requirements. Examples of such methods are power supply shutdown, dynamic power management, clock gating, and adaptive supply voltage scaling.

In a simple embodiment of power management, a system component, e.g. a particular chip, is completely separated from the power supply via an external controllable regulator during idle periods [6]. This is an effective way of avoiding unnecessary static and dynamic power dissipation in inactive components that does not complicate the design of the component to be shut down. The power manager unit (PMU) that controls the regulator is completely external and the power supply pins are the only required interface to the power-managed component. Thus, the component can be designed in the traditional way without the need for any special power management support to be implemented. Major drawbacks of this power supply shutdown approach are the following. Firstly, there is a large power-on delay, which is the time it takes for the supply voltage to stabilize after being switched on again. Secondly, the registers and other non-permanent memory cells lose their content.

Power supply shutdown can, in principle, be applied to blocks within an integrated circuit instead of to the entire chip. This, however, requires the power supply infrastructure on the chip to be modified such that the power supply nets of the different blocks are separated from each other and made accessible from the exterior via separate pins. As a consequence, power supply shutdown is restricted to chips in their entirety or to a small number of large blocks on a chip.

Complex electronic systems such as personal computers may include advanced dynamic power management (DPM) schemes. Such systems contain various power-manageable components (PMC) controlled by a PMU [8]. Each PMC provides a number of high performance, low power, and sleep modes/states. The PMU, which may be implemented in

hardware or in software, continously observes the system and puts the PMCs in appropriate states according to the actual requirements at certain points in time.

Dynamic power management is widely used in modern notebook computers and, hence, special notebook processors are designed as PMCs. This requires the instruction set, the clock network, the interrupts, etc. to be adapted to the requirements of dynamic power management. Most processors support different low power and sleep modes. In some modes, idle modules within the processors are not separated from the power supply as in the power supply shutdown approach. Instead, the respective parts of the clock network are switched off [8]. If all inputs of the modules to be switched off are registered, there is absolutely no switching activity and, hence no dynamic power dissipation in the idle modules. This technique is called global clock gating. In other modes, certain modules are actually separated from the power supply via internal switches in the power supply nets [8]. Finally, for modules which are not completely idle but also not fully utilized, the clock frequency or the supply voltage or both may be momentarily reduced.

Although designing a PMC requires a significant amount of additional design effort, the most challenging task is the development of an effective power management policy (PMP) and its implementation as PMU firm- or software [8]. This software should know about the power characteristics of all modules and be aware of the inevitable performance degradation and power overhead associated with going to and returning from the different low power and sleep modes. An effective PMP should reliably predict the idle time of a module and accurately calculate the net power reduction.

The Advanced Power Management (APM) specification was the first industry standard in the field of DPM and has only recently been replaced by the more powerful Advanced Configuration and Power Interface (ACPI) [6, 8].

Local clock gating is another popular power management technique that requires only moderate additional design effort. It is frequently used in simple processors such as DSPs, application specific processors, embedded processors and the like, but can be applied to practically any type of circuit. With local clock gating, the control signals that are used to deactivate certain parts of the clock network are locally generated in hardware. In principle, arbitrarily small subcircuits can be deactivated in this way. Since power management based on local clock gating is rather an architectural-level than a high-level technique, more details follow in Section 3.4.

A relatively new power management approach is adaptive supply voltage scaling. This is a very attractive technique for dynamic power optimization if the requirements on the performance of a chip vary continously over time. Instead of just switching off idle components of a system or idle modules on a chip, the clock frequency and the supply voltage are continously adjusted to the instantaneous performance demand. Adaptive supply voltage scaling is discussed in Chapter 4.

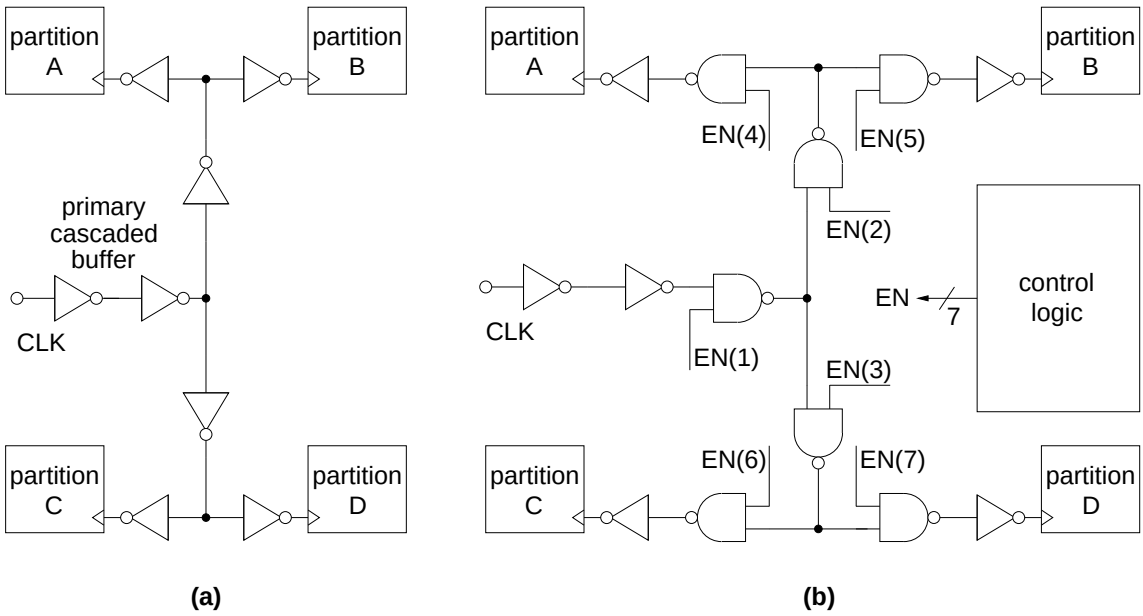

Figure 3.2: Concept of clock gating: (a) clock tree without gating elements; (b) hierarchical clock gating [112].

3.4 Architectural Optimization

The two most important methods for power optimization at the architectural level (RTL) are clock gating, which is presented in this section, and architecture-driven supply voltage scaling, which is discussed in Chapter 4. Besides clock gating, this section covers bus and state encoding and the power characteristics of arithmetic units.

Clock gating. The clock network of a synchronous digital IC normally contains clock buffers and clock nets as shown in Figure 3.2a. The entire clock network, which is frequently called clock tree, is driven by a primary buffer, and subordinate buffers are distributed across the chip. The branches of the clock tree all end at clock input pins of sequential cells such as flip-flops.

The large number of driven cells and the large total wire length bring about a large capacitive load on the clock network. Moreover, the switching activity in the clock network is usually the highest of all nets. These are the primary reasons for the large contribution of the clock tree to the total dynamic power consumption of many chips. In [96], the contribution of the entire clock network including the primary and subordinate clock buffers is quoted at 20% to 45% for different design examples. Thus, clock networks are important targets of low power design.

An effective means of reducing the power consumption in clock networks is clock gating. The concept of clock gating is illustrated in Figure 3.2b [112]. Logic gates are inserted in the clock tree in a hierarchical manner, either as replacements for or in addition to existing clock buffers. Each of these clock gating cells receives at its input pins a clock signal, which is derived from the primary clock signal CLK, and an enable signal EN, which is generated by global or local control logic, so as to activate or deactivate certain portions of the clock tree. If large portions of the clock tree are deactivated for long periods of time, the power consumption in the clock tree is significantly reduced.

Local clock gating is often used in processors, where functional units in the data-path can be deactivated when they are not needed for the execution of a particular instruction [48, 63, 69, 91, 121]. In this case, the clock enable signals are generated by the instruction decoder. If registers are placed at all inputs of the functional units, clock gating not only affects the power consumption in the clock network itself but suppresses all switching activity within the deactivated data-path units as well.

The implementation of gated clocks increases the complexity of the control logic and, hence, creates some power overhead. The overhead is acceptable if it is compensated by the power savings. The correct timing of the enable signals is the most serious issue in the design of clock gating circuitry; glitches at the clock inputs of sequential cells must be avoided in order to assure proper operation of the circuit [121].

Clock gating is often modeled in the HDL code. However, commercial synthesis tools such as BUILDGATES EXTREME (CADENCE[2]) and POWER COMPILER (SYNOPSYS[3]) are also capable of automatic implementation of clock gating.

Bus encoding. Low power bus encoding aims at reducing the switching activity and, hence, the dynamic power consumption on long multi-bit interconnects. As depicted in Figure 3.3a, bus encoding schemes generally require additional circuitry for the encoding and decoding at the transmitter and receiver side, respectively. This detracts from the overall power reduction. The effectiveness of low power bus encoding also depends on the signal statistics and the knowledge thereof. Particularly important in this respect is the correlation between consecutive data words to be transmitted.

Gray coding is often discussed in the context of instruction address encoding in microprocessor systems [34]. Normally, consecutive instructions are stored at consecutive positions in the memory, so that mostly a fixed increment is added to the program counter. If this increment is one, as for byte-addressable memory and a fixed instruction length of one byte, the Gray code may be used instead of the ordinary binary code. The advantage of the Gray code is that an increment of one changes only one bit. Since the Gray code is just a re-ordered binary code, the idea of Gray encoding can be adapted even if the standard increment is different from one. For instance, if the increment is two, as for byte-addressable

[2] http://www.cadence.com
[3] http://www.synopsys.com

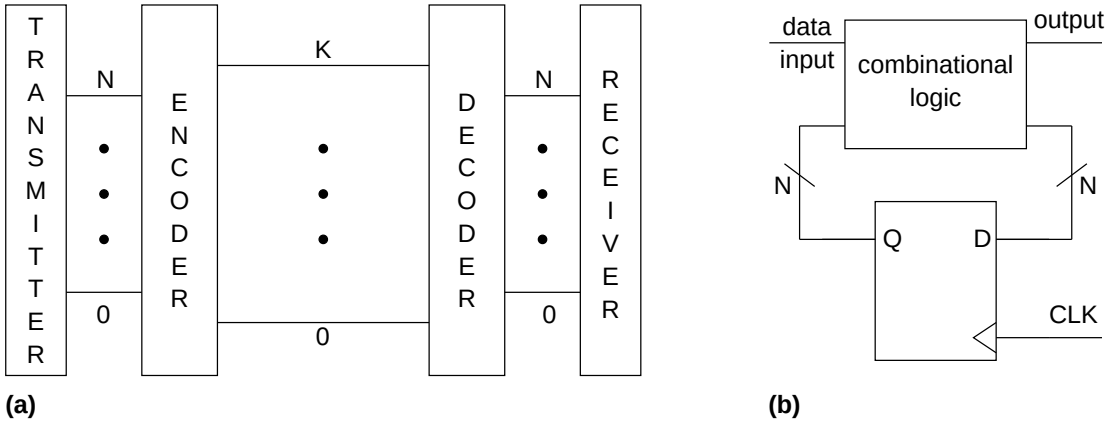

Figure 3.3: Typical applications of encoding schemes: (a) generic bus encoding architecture (redundant codes require $K > N$); (b) generic finite state machine (FSM).

memory and a fixed instruction length of two byte, the code can simply be re-ordered such that an increment of two changes only one bit.

This concept works only for the strictly sequential parts of a program; branch and jump instructions reduce the optimization potential. Also, data memory accesses detract from the optimization potential if the same address bus is used for the instruction and the data addresses. In the case of variable instruction lengths, the advantage of the Gray code vanishes because the increment is not fixed and the signal statistics are no longer predictable.

The overhead of Gray address encoding is small. If the program counter and the memory address decoder are already adapted to the optimized coding style, no extra circuitry for the encoding and decoding is needed.

If no correlation between data words exists or if the signal statistics are unknown, redundant codes may be used for reducing the switching activity. The advantages and disadvantages of redundant codes can be illustrated using one-hot coding as an example [17]. In the one-hot code of a decimal value M only the M-th bit is set to one while all other bits are zero. Consequently, regardless of the signal statistics, the number of switching bits per cycle is two when the data changes, and zero otherwise. The drawback is that representing 2^N numbers requires $K = 2^N$ bits as opposed to N bits required for the ordinary binary coding. The result is an unacceptable overhead for routing, encoding, and decoding.

Bus inversion coding (BIC) is an example of redundant bus encoding with low overhead [17, 100, 121]. In a first embodiment, BIC requires only one additional signal line. The basic idea is to invert a data word prior to transmission if this reduces the number of switching bus lines. The additional line (polarity line) is used for signaling to the receiver whether the data word has been inverted or not. Switching events on this line must, of course, be taken into account when deciding on the polarity of transmissions.

The effectiveness of BIC degrades with increasing bus width. Therefore, broad busses should be split into narrow slices with separate en-/decoders and a separate polarity line for each slice. A maximum switching activity reduction of 25% can be achieved by splitting an N-bit bus into 2-bit slices at the cost of $N/2$ extra wires [121]. This overhead is small compared with one-hot coding. Nevertheless, it is often unacceptable. Thus, four or eight bit are more realistic choices for the width of the slices. The overhead caused by the decoder is small. The encoder, however, can be quite complex and must be taken into account when weighing up advantages and disadvantages of BIC [17, 99].

Another way of dealing with a lack of knowledge of the signal statistics is adaptive bus encoding, where the incoming data stream is continously observed and the en-/decoding rules are adapted to the varying statistical properties of the data stream. Recently, an adaptive bus encoding scheme, which is based on the probability based mapping (PBM) technique, was presented [62]. With PBM, the switching activity on the bus is minimized by minimizing the number of ones to be transmitted. Frequently occurring data words are mapped to code words that contain a small number of ones. A one is transmitted over the bus by inverting the state of the respective bus line. For transmitting a zero, the state of the bus line is maintained. The PBM technique uses a non-redundant data representation and, thus, requires no additional bus lines.

The code computation circuitry implemented at both ends of the bus continuously determines a probability of occurrence for each data word in the data stream, computes a new mapping rule in certain intervals, and writes the rule to look-up tables. The PBM scheme can effectively reduce the switching activity if certain data words occur much more frequently in the data stream than others. If, on the other hand, all data words are uniformly distributed, the benefit of PBM vanishes. While a static PBM scheme, where the code mapping rule is optimized for a specific data stream, often yields bad results when used on other data streams, the adaptive PBM scheme can be successfully applied to different data streams or to data streams that exhibit varying statistical properties.

Adaptive bus encoding schemes require complex en-/decoding circuitry. The resulting power and area overheads may predominate the possible power savings. Another problem with adaptive bus encoding, which has not been completely solved yet, is the synchronization of the adaption mechanisms at the transmitter and receiver sides of the bus.

State encoding. Finite state machines (FSM) are composed of the state registers and combinational logic which computes the output signals and the next state on the basis of the input signals and the current state as depicted in Figure 3.3b.

The behavior of FSMs is often described using state transition diagrams, which are directed graphs where the nodes represent states and the edges describe transitions between the states. The starting point and the endpoint of a transition are called the current state and the next state, respectively. An example of such a diagram is given in Figure 3.4a. This FSM has three states that are denoted S1, S2, and S3. For the actual hardware implementation

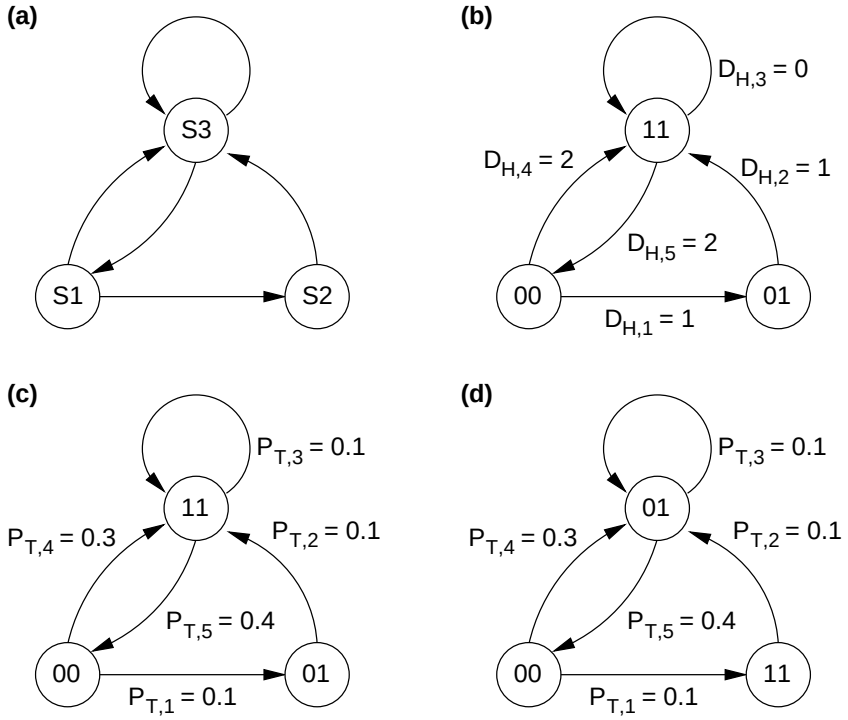

Figure 3.4: Low power state encoding [121]: (a) state transition diagram representing an FSM with three states; (b) random binary state encoding and Hamming distances; (c) transition probabilities; (d) low power binary state encoding.

of an FSM, such abstract notations of state names must be translated to some sort of binary state encoding. In the example, S1 could for instance be translated to '00', S2 to '01', and S3 to '11', as shown in Figure 3.4b. This binary state encoding requires two registers for representing the three states. If a transition occurs, some of the state registers change their logic state from '0' to '1' or vice versa, which causes some circuit capacitances to be charged or discharged. The aim of low power state encoding is to minimize the total switching activity and, hence, the dynamic power consumption due to state transitions.

In one approach to low power state encoding, the cost function is the sum of the Hamming distances $D_{H,i}$ between all possible pairs of current and next states. The Hamming distance between two states is the number of bits that are different in the binary numbers representing these states. In Figure 3.4b, the total Hamming distance is six. In another approach, each edge in the graph is weighted with a number in the range of zero to one that describes the probability $P_{T,i}$ of that particular transition to occur. The cost function is modified such that the Hamming distance $D_{H,i}$ associated with the i-th pair of nodes is multiplied by the weight $P_{T,i}$ assigned to the respective transition. This cost function yields total switching activities of 1.6 and 1.0 for the original and the modified encodings depicted in Figures 3.4c and 3.4d, respectively.

Obviously, 38% less switching activity in the state register bank can be expected if the states are encoded according to Figure 3.4d. This, however, does not necessarily mean that the power consumption is 38% lower, too, since an impact of the state encoding on the complexity of the combinational logic and on the switching activity therein is not considered in the cost function. In fact, the total power and the area of the FSM may increase, if a certain change of the state encoding, which reduces the total number of toggling bits in the register bank, requires a more complex combinational logic. Experimental results presented in [121] show tremendous variation in terms of area and power consumption of a particular FSM at a given performance depending on the encoding of the states. It seems that larger state machines tend to consume more power. However, the correlation between state encoding, power, and area is still not well understood. Thus, the real effects of state re-encoding are difficult to predict. This lack of understanding is a major obstacle for the development of efficient tools for power-conscious FSM design.

Some commercial logic synthesis tools such as SYNPLIFY PRO (SYNPLICITY[4]) and DESIGN COMPILER (SYNOPSYS[5]) provide basic FSM design and optimization capabilities targeting only the circuit area. Simultaneous optimization for area and power consumption is not yet supported and, hence, is left to the designer. However, the complexity of FSMs in real designs is mostly too high for manual optimization. A practical solution to this problem is to manually encode only a small subset of states that covers the edges that contribute the most to the expected switching activity and to leave the encoding of the majority of states to the synthesis tool with the objective of minimizing the area.

Low power arithmetic units. Arithmetic units such as adders and multipliers are critical building blocks in processors and many data-path-dominated ASICs. A variety of concepts for the implementation of such modules can be found in the literature [120]. While, in the past, the design of arithmetic units was driven by the need for sufficient performance at minimum area, their power consumption can now no longer be ignored.

Callaway and Swartzlander have investigated and compared different types of parallel 16-bit adders and multipliers [16]. Some results of their work are summarized in Tables 3.1 and 3.2. Evidently, faster implementations mostly require larger area. For the adder circuits listed in Table 3.1, shorter delay and larger area also translate to higher dynamic power consumption. This is different for the multipliers listed in Table 3.2; the second fastest circuit (Wallace tree) consumes the least dynamic power, while the slowest implementation (array) results in the highest power consumption.

The power-delay product (PDP) given in the tables is a possible measure of the trade-off between performance and power. In this respect, the minimum PDP values mark the most efficient implementations of adders and multipliers (variable block width carry skip adder

[4] http://www.synplicity.com
[5] http://www.synopsys.com

Circuit type	Delay	Power	PDP	Area
Ripple carry	1	1	1	1
Constant block width carry skip	0.56	1.06	0.59	1.27
Variable block width carry skip	0.44	1.29	0.57	1.88
Carry look ahead	0.44	1.59	0.70	2.04
Carry select	0.36	2.24	0.81	3.38
Conditional sum	0.41	3.18	1.30	4.38

Table 3.1: Delay, power consumption, PDP, and area of 16-bit adders normalized to the delay, the power, the PDP, and the area, respectively, of the ripple carry adder [16].

Circuit type	Delay	Power	PDP	Area
Array	1	1	1	1
Split array	0.68	0.87	0.59	1.43
Wallace tree	0.58	0.74	0.43	1.93
Modified booth	0.49	0.95	0.47	2.02

Table 3.2: Delay, power consumption, PDP, and area of 16-bit multipliers normalized to the delay, the power, the PDP, and the area, respectively, of the array multiplier [16].

and Wallace tree multiplier). On the basis of this perception, Wallace tree multipliers were, for instance, built into certain StrongARM low power processor derivates [11].

For a detailed discussion of the structure and the functioning of the different types of adders and multipliers considered in this comparison see the literature [15, 16, 120].

3.5 Logic Level

Standard-cell-based design at the logic level includes logic synthesis, placement, and routing. Logic synthesis can be further divided into technology independent and technology dependent optimization steps.

Technology independent optimization. Technology independent optimization requires the combinational part of the original design to be separated from the sequential elements. The combinational logic is described in the form of Boolean equations, and the optimization methods operate on these equations. Traditionally, the goal is to find an area efficient,

multi-level representation of the combinational logic under timing constraints [9]. A common measure of the area of Boolean networks is the total number of literals[6] in the factored form of the equations [49]. Therefore, the traditional objective of technology independent optimization is the minimization of the total number of literals. This is usually done with algebraic logic restructuring techniques, e.g. extraction, substitution, factorization, and Boolean minimization.

Extraction is the process of identifying a common sub-function of several Boolean equations, introducing a new equation that assigns the common sub-function to a new internal variable, and substituting the common sub-function in the original equations with the new variable [9]. Substitution means substituting a sub-function of a Boolean equation with an existing internal variable [9]. Substitution is applicable if internal variables exist that represent sub-functions of other equations. Another important technique is factorization [9]. The Boolean expression $a \cdot c + a \cdot d + b \cdot c + b \cdot d$ for instance, can be transformed to $(a+b) \cdot (c+d)$. In this example, factorization reduces the number of literals, which is one purpose of the technique. The other purpose is the computation of the cost which is often based on the factored form of the equations, as mentioned above.

The same methods can be used for technology independent dynamic power optimization, if the cost function is modified [49]. The cost may be computed as the total sum of the switching activities associated with all literals. This requires the switching activities of the primary inputs to be specified. The switching activities associated with internal variables and primary outputs are then computed by propagating the switching activities at the inputs through the Boolean network using zero delay models for the Boolean operations.

Boolean minimization is the process of minimizing Boolean equations using the rules of Boolean algebra, e.g. $a + \bar{a} = 1$, and taking into account any don't care conditions [9]. ESPRESSO is a de-facto-standard algorithm for computer aided Boolean minimization of two-level Boolean networks targeting the area of the resulting circuit [9, 39, 80]. Similar methods can be applied to the set of equations that describes a multi-level Boolean network taking into account additional don't care conditions [82]. Power-aware Boolean minimization, however, requires modification of these methods, as discussed in [49].

Finally, the optimized Boolean network is prepared for technology mapping in a step called technology decomposition [9, 49]. A set of primitive Boolean functions such as two-input NAND and NOT is chosen. The Boolean equations are then converted to a graph where each node in the graph is restricted to one of the primitive functions. This process is called technology decomposition and the result is called the subject graph. This graph is the input to technology mapping, the first step in the technology dependent phase of logic synthesis. The quality of the mapping solution depends on the structure of the subject graph. According to [49], a subject graph that minimizes the sum of the switching activities associated with its internal nodes is a good starting point for low power technology mapping.

[6] A literal is a negated or not negated instance of a variable in a Boolean expression. In multi-level Boolean networks, the variable may be an internal variable or represent a primary input.

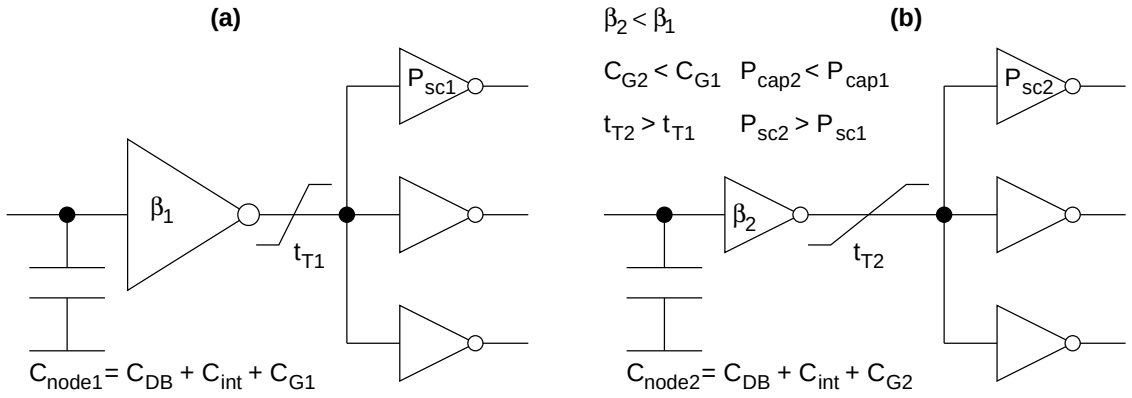

Figure 3.5: Gate sizing for dynamic power optimization: (a) large driver, large capacitance, short transition time; (b) smaller driver, smaller capacitance, longer transition time.

Technology dependent optimization. The technology dependent phase of logic synthesis starts with a step called technology mapping or cell binding [9, 49]. In this step, the functionality of each library gate is represented by a graph where each node is restricted to the primitive Boolean functions considered in technology decomposition. These graphs are called pattern graphs. Technology mapping is the process of finding a minimum cost covering of the subject graph by choosing from the collection of pattern graphs that represents the standard cell library. Again, switching activities should be considered when computing the cost in order to obtain a low dynamic power mapping solution [49].

The technology mapping is followed by a post-mapping optimization phase. An important technique applied at this stage is gate sizing. In addition to gate sizing, local transformations are used for altering the structure of the circuit without changing its functionality. Typical examples of local transformations are buffer insertion, complex gate composition and equivalent pin swapping.

Gate sizing can affect the dynamic power consumption P_{dyn} in different ways as illustrated in Figure 3.5. Down-sizing, i.e. replacing a cell with a functionally equivalent cell composed of smaller transistors that have smaller gate input capacitances C_G, primarily aims at reducing C_{node} and, thus, P_{cap} at the input nodes of the sized cell [49, 121]. In addition, smaller transistor dimensions reduce the short-circuit and the subthreshold currents in the sized cell, thus reducing P_{sc} and P_{sub}. On the other hand, down-sizing increases the signal transition time t_T at the sized cell's output, which in turn increases P_{sc} of the cells driven by the sized cell. For this reason, minimizing the size of cells in non-timing-critical paths does not always result in the lowest dynamic power consumption. At heavily loaded nodes that exhibit very large t_T, up-sizing may lead to an overall lower P_{dyn} [12].

Alternatively, extra buffers can be inserted at heavily loaded nodes in order to shorten t_T, as shown in Figure 3.6 [21]. This reduces P_{sc} at the gates driven by the inserted cell. However, the extra cell introduces extra capacitances and extra short-circuit currents. These

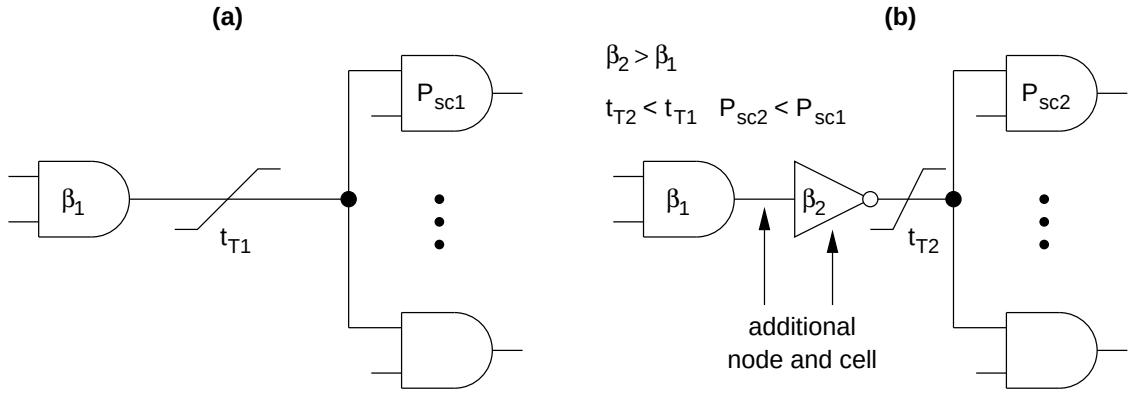

Figure 3.6: Buffer insertion for short-circuit power optimization [21]: (a) long transition time at heavily loaded net; (b) extra buffer shortens transition time.

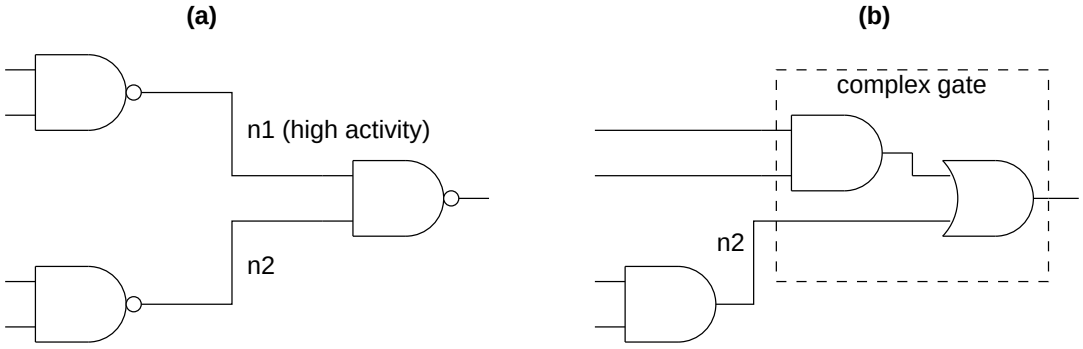

Figure 3.7: Complex gate composition for capacitive power optimization [11]: (a) high activity net connects separate cells; (b) high activity net hidden in complex gate.

overheads must of course be small in comparison with the reduction in short-circuit power at the driven gates in order to make this buffer insertion technique feasible.

Standard cell libraries contain so-called complex gates which combine several simple gates in one cell. Complex gate composition replaces a group of simple gates in a gate-level netlist with an equivalent complex gate as depicted in Figure 3.7 [11, 21]. As a result, some nets no longer connect separate cells. Instead, these nets connect devices within a complex cell which can be accomplished with shorter wires that have less capacitance. This reduces P_{cap}, especially if many high activity nets can be hidden in complex cells.

Another optimization technique, which is called equivalent pin swapping or pin ordering, exploits the different power characteristics of functionally equivalent input pins of the same library cell. These differences in the power characteristics can be due to different input pin capacitances, which leads to different P_{cap} at the different input nodes. Another possible reason is the exact position of the devices connected to a particular input pin, i.e. the

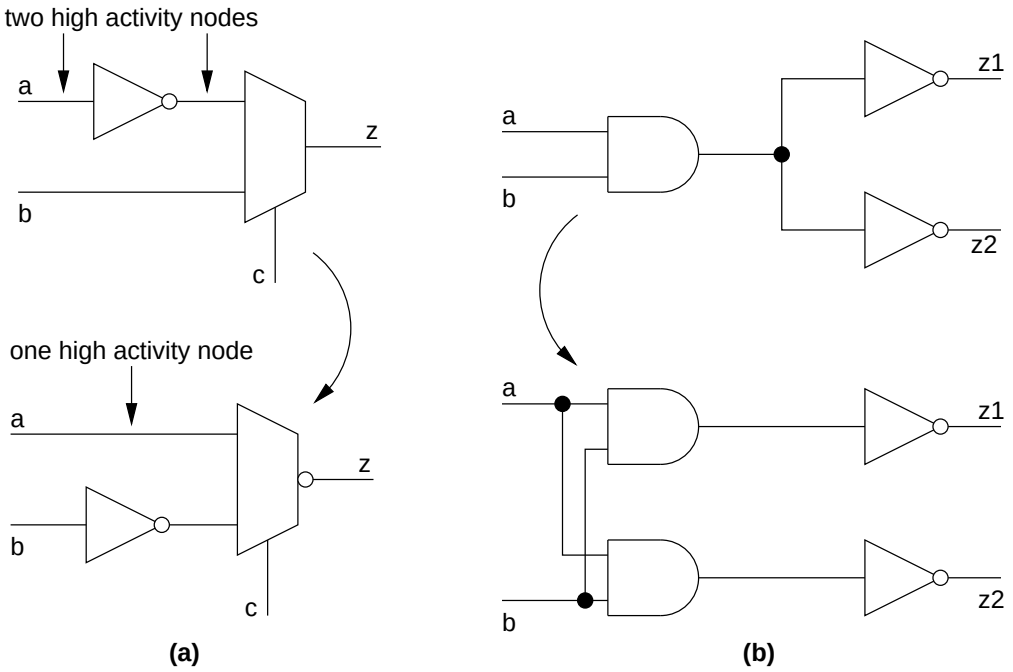

Figure 3.8: Local transformations for post-mapping logic restructuring: (a) phase assignment [11]; (b) cell duplication [121].

cell-internal circuit structure, which affects the total cell-internal capacitance charged or discharged during a transition of the input node. With pin swapping, high activity nets are connected to power-efficient input pins with priority [11, 21].

A variety of other local transformations for logic restructuring after technology mapping was proposed [88, 93, 121]. Two examples are illustrated in Figure 3.8. In Figure 3.8a, one of the two high activity nodes is eliminated by inverting the input and output signals of the multiplexer, thus reducing the dynamic power consumption. This technique is often called phase assignment. Duplication of cells, as shown in Figure 3.8b, can be used for splitting paths. The opposite of complex gate composition, i.e. complex gate decomposition, serves the same purpose. These techniques may not directly reduce the power consumption but enable other transformations that eventually lead to lower power.

Technology dependent power optimization is supported by commercial tools, for instance by POWER COMPILER (SYNOPSYS[7]) and BUILDGATES EXTREME (CADENCE[8]).

Placement and routing. The traditional objective of placement is to arrange all cells on the chip in such a way that the total wire-length after routing is minimized and, thus, the

[7] http://www.synopsys.com
[8] http://www.cadence.com

area is minimal. Since the actual wire-lengths are unknown at this stage, estimates are used for computing the cost function. For power-driven placement, the estimated wire-lengths should be weighted with the switching activities, so that high activity nets are given priority. This way, the total switched capacitance, which determines P_{cap}, is minimized instead of the total wire-length, which affects the area. In principle, power-driven placement can be carried out with the same algorithms as conventional placement if the cost function is modified appropriately [24].

Routing is the process of making electrical connections between pins of placed cells. In conventional routing, the objective is to minimize the total wire-length. The limitations on the routing resources, i.e. the routing area, the number of metal layers, and the number of feed-throughs between these layers, frequently lead to region congestion. Therefore, it is usually not possible to minimize the length of every single wire. At the beginning of the routing process many resources are available and most wires can be realized with minimum length. As the routing process progresses, congestion problems become more likely and wire-lengths increase. For this reason, critical nets should be routed first. Again, for power-driven routing, the priorities of nets can be determined from the switching activities, so that high activity wires are kept short [24].

The coupling capacitances between neighboring wires are significant sources of power consumption. Therefore, power-driven routing should not only address the wire-length but also reduce the coupling capacitances between high activity wires by increasing their spacing [24].

3.6 Transistor Level

The standard cell ASIC design style is based on the concept of reusing pre-designed logic gates, 1-bit adders, flip-flops, etc. that are available in so-called standard cell libraries. The following paragraphs cover various aspects related to the development of low power standard cell libraries.

Logic styles. Logic gates can be dynamic or static, i.e. with or without clock control. Dynamic logic is generally faster and, hence, well suited to highest performance circuits. However, the power consumption of dynamic logic is larger than that of static logic because of the additional capacitive load at the clock network(s) and because of unnecessary precharging and discharging of nodes [126]. Moreover, standard tools used for logic and layout synthesis do not support dynamic logic design.

The conventional static CMOS logic gates built from n-channel pull-down and p-channel pull-up networks are easy to design, have good driving capabilities which allows high performance, and have good noise margins which makes the circuits robust even at low supply voltages. Static logic gates exploiting cross-coupled p-channel transistors, e.g. cascode

voltage switch logic gates, have larger delays and may consume equal or larger amounts of power [61, 106]. Moreover, such gates are difficult to design. Particularly, the design of cells with larger driving strengths is impractical in such logic styles. A third class of static logic, namely the pass transistor logic (PTL), appears to have little or no advantage over the conventional static CMOS gates regarding the power consumption. Moreover, the performance and the robustness of PTL at low supply voltages are insufficient [61, 106, 126].

For the reasons stated above, only the conventional static logic style can be found in standard cell libraries, except for some pass transistor or transmission gate structures used in XOR gates, multiplexers, flip-flops or full adders.

Combinational cells. Standard cell libraries typically contain cells having up to eight inputs. Larger numbers of inputs result in unfeasibly large numbers of transistors connected in series and in parallel. Many transistors connected in series limit the low voltage operation and have either a large total series resistance or large gate capacitances. Many transistors connected in parallel introduce a large total drain diffusion capacitance at the output. Finally, the body effect increases the threshold voltage of transistors connected in series. These effects lead to poor performance [121].

Another important aspect regarding the low power library development is the selection of Boolean functions to be implemented. The number of different Boolean functions of N input variables is $M = 2^{2^N}$. For three inputs, for instance, M is 256 and for four inputs M is 65536. It is obvious that only a small collection of all these possible functions can actually be included in a standard cell library. Unfortunately, there is a lack of theoretical analysis of the problem of which functions to implement. Therefore, the actual selection of functions and, hence, types of cells is usually based on human intuition and experience. Typical industrial libraries contain non-inverting buffers, inverters, (N)AND gates, (N)OR gates, X(N)OR gates, (N)AND-(N)OR complex gates, multiplexer, 1-bit half and full adders and similar cells [121].

Low power libraries should provide a sufficient number of complex gates, i.e. (N)AND gates, (N)OR gates, and cells with integrated input inverters. This enables effective complex gate composition for power and area reduction. Also, every type of cell should be provided in sufficiently many different sizes (driving strengths), including minimum sized cells and cells with asymmetrical timing characteristics due to reduced p-channel widths, in order to enable effective gate sizing for timing, power and area optimization. Particularly, non-inverting buffers and inverters, which are frequently used to form optimized cascaded buffers for driving large loads, should be available in a large number of different sizes [3].

Flip-flop cells. Other than dynamic logic gates, dynamic flip-flops may be more power efficient than their static counterparts. This is because dynamic flip-flops can be realized with less transistors and the load presented to the clock network is smaller [105]. However, logic states stored in dynamic circuits need to be periodically refreshed, which prevents

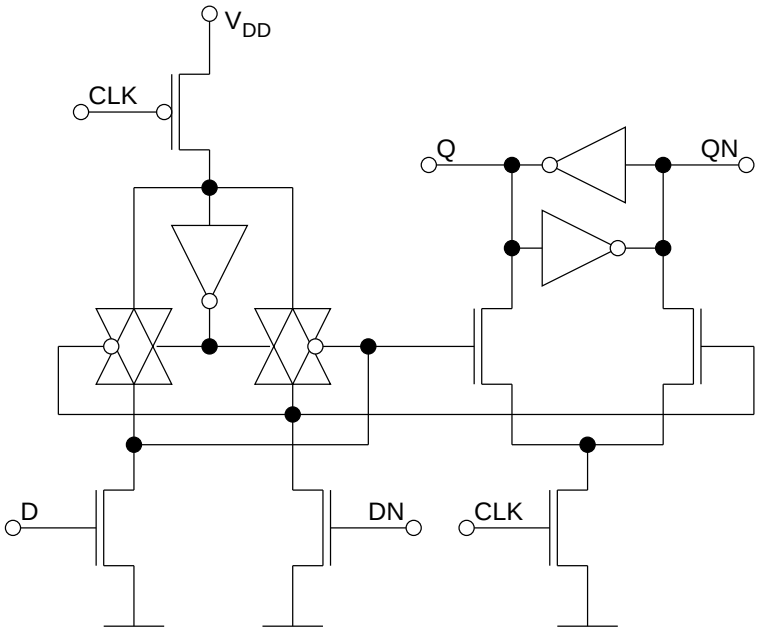

Figure 3.9: Low Power D-flip-flop circuit.

dynamic flip-flops from being disabled using clock gating or other means. Moreover, dynamic circuit design is not supported in standard-cell based ASIC design methodologies, as mentioned above. Therefore, only static flip-flops exist in standard cell libraries.

A widely-used static flip-flop structure is shown in Figure 6.8. The transmission gates are sometimes replaced with tristate buffers, but with this exception, most flip-flop cells in commercial standard cell libraries have this basic structure in common. A disadvantage of this circuit is the large effective load presented to the clock network. This load includes the capacitances that are charged and discharged by the internal clock buffer.

Low power flip-flop designs aim primarily at reducing the load presented to the clock network. The circuit depicted in Figure 3.9, for instance, has only two transistors driven by the clock input pin. For low to medium switching activity at the data input, this flip-flop consumes less power than the standard flip-flop [3]. At the same time, its delay is significantly shorter. A similar flip-flop comprising a modified master latch is described in [29]. This circuit consumes less power even for high switching activity at the data input.

Cell layout optimization. The means of optimizing the standard cell layouts for low power are limited. Merely optimizing the intra-cell interconnects and the gate structure of very wide transistors can be worthwhile. Large transistors have large drain/source diffusion capacitances if they are realized with a longitudinal gate structure. As explained in [17], the drain capacitance is reduced if the gate is laid out with a finger or ring structure. This

technique can be applied, for instance, to large buffer and inverter cells or to logic cells with large output buffers. For long interconnects within large cells, such as complex flip-flop cells, it is worth considering the area-specific capacitance of different interconnect materials such as polysilicon and metal in order to minimize the wire capacitance.

3.7 Summary, Comments, and Conclusions

As mentioned at the beginning of this chapter, low power ASIC design methodologies should include power optimization at all levels of abstraction. Which particular techniques to include in a real-world methodology is determined by their effectiveness, stage of development, versatility, and suitability for automation. These criteria lead to the following assessment of the low power design methods discussed in this chapter.

The implementation of power management for static or dynamic power reduction or both is a must, unless it is not applicable to the target application. Dynamic power management can be very effective but requires a tremendous design effort. Therefore, DPM is restricted to the design of complex systems such as personal computers and parts thereof. Clock gating has also proven to be effective and, fortunately, its implementation is simple compared with DPM. Local clock gating is even supported by commercial tools. This technique is state-of-the-art in ASIC design and should be used whenever possible. Regarding the focus of this study, it is important to investigate the impact of clock gating on the effectiveness of voltage scaling in the clock network. This is addressed in Chapter 8.

Low power bus encoding is a very difficult and conflicting problem. Simple schemes like Gray and one-hot coding are either lacking versatility or are too expensive because of tremendous overheads. Static PBM works well only for the data stream it was designed for and is, thus, only slightly more versatile than Gray coding. Adaptive PBM creates large overhead and suffers from unsolved technical problems. At present, only BIC appears to be useful for a broader range of applications. Low power state encoding is complex, not well understood and only partially supported by tools. However, if a small subset of transitions can be identified as main contributer to the dynamic power consumption of a particular ASIC, it can be worth encoding the respective states manually while leaving the encoding of the majority of states to a synthesis tool. An impact of bus and state encoding on the effectiveness of the methodology proposed in this study is not expected.

The design of optimized arithmetic units from scratch is carried out only in the full-custom design of high performance components such as general purpose microprocessors. For the design of ASICs, technology-independent macro block libraries are available, that contain a variety of pre-designed arithmetic units. Logic synthesis tools revert to these library elements when processing RTL design descriptions subject to timing, power and area constraints. If the constraints cannot be met this way, optimized HDL modeling of arithmetic units can be applied instead of using arithmetic operators in the HDL code. The latter approach is particularly suitable for the design of critical units in the data-path of application

specific processor cores; the arithmetic units can be adapted to the target application while preserving synthesizability and, thus, independency of the target fabrication technology. Just as for the aforementioned encoding schemes, an impact on the effectiveness of the methodology proposed in this study is not expected.

Logic synthesis is fully automated and relies on standard tools. These tools do not support power optimization in the technology independent phase of logic synthesis. Technology dependent optimization using gate sizing, buffer insertion, complex gate composition, equivalent pin swapping, phase assignment, etc. is state-of-the-art and should be used in any case. About 10% to 20% dynamic power reduction can be expected. These techniques directly compete with the logic-level voltage scaling approach that is in the focus of this study. Therefore, the proposed methodology assures that the effect of state-of-the-art power-driven logic synthesis is taken into account in all investigations.

Placement and routing are also automated and are also carried out using standard tools. In existing design methodologies, the area is usually the only optimization criterion. Timing-driven placement and routing are possible but are not yet standard. Power-driven placement and routing are still under development and cannot be carried out with existing tools.

Regarding the library development, many semiconductor vendors avoid the effort to develop completely new libraries. Instead, existing libraries are adapted to newer technology generations with minimum effort. Some companies, e.g. ARTISAN COMPONENTS[9], claim that their libraries are optimized for low power design. However, neither is there any evidence, nor is any information available on how this was achieved.

The techniques discussed in this chapter aim at power optimization through power supply shut-down and through optimization of circuit and device parameters such as the switching activity, the device and interconnect capacitances, the signal transition times, and the effective transconductance. Other important parameters are the supply and threshold voltages. However, the simple concept of global supply voltage minimization driven by pipelining or parallelization is usually the only available voltage scaling option. More advanced supply and threshold voltage scaling approaches, although generally considered promising, are not state-of-the-art. The broad field of voltage scaling is covered by the following chapter.

[9] http://www.artisan.com

Chapter 4

Supply and Threshold Voltage Scaling

4.1 Conventional Voltage Scaling and its Limitations

Supply voltage (V_{DD}) scaling is an effective means of dynamic power (P_{dyn}) optimization because of the non-linear dependence of P_{dyn} and V_{DD} (see Equations 2.19 and 2.22). The supply voltage can, however, not be scaled down without limits.

The first question to be answered is that of possible malfunction at low supply voltages. A common condition for a logic gate being properly functioning is a gain[1] significantly larger than one in order to guarantee level restoration from one stage to the next. The ultimate goal is to have per-stage gains large enough to keep the logic levels at all circuit nodes within the specified noise margins. It was shown theoretically that this condition can be satisfied even at extremely low supply voltages of a few hundred millivolts [71, 79, 107]. First practical examples of circuits operating at supply voltages as low as 0.2 V were published 30 years ago [108]. Thus, the possibility of malfunction has not yet become a real limit.

Practically relevant limits to low voltage operation arise from timing constraints. According to Equation 2.15, the gate delay t_D and, hence, the performance of a circuit inevitably degrade, if the supply voltage is reduced while the threshold voltage remains at the same level. This problem can be overcome by simultaneous supply and threshold voltage scaling. Unfortunately, low threshold voltages cause excessive static power consumption P_{stat}, as discussed in Section 2.3.3. This means that aggressive supply and threshold voltage scaling eventually leads to a minimum of the total power consumption P_{tot}, which defines an optimal pair of supply and threshold voltage values. If the voltages are scaled beyond this minimum, P_{tot} increases again because of the exponentially rising subthreshold currents.

In practice, the optimum is usually not well defined because of unavoidable supply and threshold voltage uncertainties that can be due to temperature and process variability, short-channel effects and non-ideal supply voltage regulation [98]. These voltage uncertainties

[1] The gain is the absolute value of the slope of the voltage transfer characteristic.

result in delay and power variations and must be taken into account when determining the nominal voltage values. Otherwise, the timing constraints may be violated or excessive static power may result or both [102].

Because of the exponential characteristic of P_{stat} (see Equation 2.24), threshold voltages below the optimum lead to a significantly larger total power consumption. This will become even more of a problem in future fabrication processes, as the nominal threshold voltages will continue to be scaled down while the threshold voltage uncertainties are not expected to scale accordingly.

Since threshold voltages below the optimum have a more serious impact on the total power consumption than somewhat increased supply voltages, the nominal voltages must be set to values above the optimum. In other words, given a certain target performance, threshold voltage uncertainties set lower limits to both the nominal threshold and supply voltages. As a result, all circuits that do not exhibit worst-case parameters consume more power than necessary because of the unnecessarily large supply voltage.

It follows from the above arguments that simultaneous supply and threshold voltage scaling subject to timing constraints yields an optimal voltage pair that minimizes the total power consumption. Sophisticated optimization approaches yield optimal solutions taking into account various different sources of voltage uncertainties [30, 35]. The optimum solution, however, depends on the application and the desired performance, and designers are usually not free to choose supply and threshold voltage values depending on the actual requirements of the application. This makes the use of optimal voltages impossible in most practical cases.

The design of standard-cell-based ASICs is typically subject to close restrictions regarding the choice of supply and threshold voltages. Many manufacturing processes provide only one fixed threshold voltage. If two types of transistors, one with low and the other one with high V_t, are needed, a special dual threshold voltage (DTV) process is required. However, the standard cells provided with existing libraries are often designed assuming that only one threshold voltage is available. Even the supply voltage can often not be chosen arbitrarily, because standard cell libraries are usually characterized and, thus, qualified for only one, sometimes for two, different supply voltages.

In the following sections, various voltage scaling techniques for timing-constrained static and dynamic power optimization are discussed. The basic ideas behind these techniques are the following. Firstly, decreasing the logic depth in the critical path and introducing parallelism reduces the optimal supply voltage value. Secondly, minimizing voltage uncertainties enables the nominal voltage values to be set closer to the optimal values. Finally, making supply and threshold voltages variable or having multiple fixed supply and threshold voltage values available introduces more flexibility regarding power-delay trade-offs into the design process.

4.2 Critical Path Relaxation for Low Voltage Operation

The basic idea of the techniques discussed in the following paragraphs is, firstly, to shorten the critical path through pipelining or retiming and, secondly, to relax the constraint on the critical path delay through parallelization. The goal is to create timing slack that can be exploited by supply voltage scaling without degrading the overall circuit performance. These principles can be applied at different levels of abstraction.

Algorithmic speed-up transformations. Algorithmic transformations such as loop unrolling, algebraic transformations, pipelining or retiming are usually applied to data flow graph (DFG) representations of algorithms [18]. A DFG is a directed graph, where the nodes correspond to operations and the edges describe the data flow between nodes. The edges may be weighted with delay units that are abstract representations of memory or states. In a DFG, a path is a set of operations connected by non-weighted edges. Paths always start at an input of a DFG or at the endpoint of a weighted edge. The endpoints of paths are either outputs of a DFG or starting points of weighted edges. The total delay of a path is the sum of the execution times of all operations along that path.

Loop[2] unrolling is a means of introducing parallelism at the algorithmic level. When architectural pipelining or parallelization are not immediately applicable, loop unrolling followed by other transformations is often the only way of creating slack in the critical path. In an N-fold parallel design, N data samples are processed simultaneously by N identical hardware units. Given a certain target throughput[3], the processing of a single sample is allowed to take N times longer than in a non-parallel design, where all samples are processed sequentially by the same hardware unit. Thus, the processing units in the parallel design may be operated at a lower supply voltage leading to reduced dynamic power.

In addition, loop unrolling often enables other algorithmic transformations that were not applicable before. Pipelining and retiming can be used for shortening the delay in the critical path. At the algorithmic level, pipelining and retiming mean to introduce additional and re-order existing delay units in a DFG. In other words, pipelining and retiming are means of path splitting and path balancing, respectively. Even algebraic transformations such as distributivity and constant propagation can sometimes reduce the delay in the critical path. After shortening the longest path, the supply voltage can be further reduced.

Parallelization and pipelining always increase the circuit complexity. For instance, additional processing units, registers, and multiplexers are needed. This becomes clearer in the discussion of architecture-driven supply voltage scaling below. The additional hardware increases the effective switched capacitance and, thus, creates some power overhead that

[2] A loop is an iteration in a recursive algorithm.

[3] Throughput is the rate at which a signal processing system consumes input data samples and produces output data samples.

detracts from the power savings. Finally, while the throughput is constant, the latency[4] inevitably increases.

In principle, algorithmic transformations can be automated in high-level synthesis which was demonstrated through a number of non-commercial tools. While most of these tools were developed for optimizing performance and area, the cost function implemented in HYPER-LP also covers the power consumption [18]. However, high-level synthesis has not yet found broad acceptance in the ASIC design community and, hence, such tools are usually not available in state-of-the-art design flows.

Architecture-driven supply voltage scaling. Parallelization and pipelining carried out at the RTL are popular means of optimizing the performance of synchronous data-path architectures. Both techniques relax the delay constraints on critical paths. The additional slack can then be exploited for higher data throughput at increased clock frequencies. However, if higher performance is not required, the increased slack in the critical paths can be exploited for reducing the power consumption by lowering the supply voltage. This approach is known as architecture-driven supply voltage scaling [17]. As opposed to algorithmic transformations, architecture-driven supply voltage scaling is not applied to DFGs but to RTL designs composed of registers, processing units, multiplexers, and the like.

With parallelization, the constraint on the delay of the critical path is relaxed while the logic depth remains unchanged. For this purpose, the block that contains the critical path is implemented N times, so that N data samples can be processed simultaneously. As explained before, the parallel architecture can be operated at a lower supply voltage in order to reduce the dynamic power consumption without degrading the throughput.

An obvious overhead introduced by parallelism is caused by the additional processing units. Besides, each of these units needs dedicated input registers. Finally, a multiplexer is required for passing the outputs of the parallel blocks sequentially to the output register. A trade-off has to be made between power savings due to voltage scaling and power overhead due to the additional circuitry.

With pipelining, the block that contains the critical path can be split into several less complex blocks connected in series. This is accomplished by additional register stages inserted along the logic paths between the existing input and output registers. Thus, a new critical path with reduced logic depth is created. The slack generated thereby can be exploited for dynamic power reduction through supply voltage scaling without degrading the throughput.

The additional registers represent some overhead because of their internal power consumption, the additional load presented at the clock network, and the required chip area. Just as in the case of parallelization, a trade-off has to be made between the power savings due to voltage scaling and the power overhead due to the additional circuitry.

[4] Latency is the time it takes an input data sample to proceed through a signal processing unit before the valid result appears at the output.

Regarding the optimization potential, both parallelization and pipelining are known to be equally effective [17]. Also, the latency increases equally for both methods. However, the overhead created by pipelining is usually much less significant compared with the overhead that comes with parallelism. Therefore, pipelining should be preferred whenever possible, and parallelization can be used if the data-path is not suitable for pipelining.

In principle, pipelining and parallelization can be automated in high-level synthesis. However, high-level synthesis has not yet become an integral part of ASIC design flows. Therefore, both techniques have to be applied manually in the architecture development (RTL modeling) phase.

Parallelization at the logic level. The concept of global supply voltage scaling (GSVS) enabled by parallelization can be applied even at the logic level. This means nothing else than a trade-off of area against delay in the technology independent phase of logic synthesis (see Section 3.5) by reducing the logic depth, i.e. the number of levels in the logic.

Just as at the higher levels of abstraction, logic-level parallelization shortens the critical path. This effect can be reinforced using gate up-sizing at the same time. The timing slack created thereby can then be exploited for dynamic power reduction through supply voltage scaling while keeping the overall circuit performance unchanged.

An increase in the degree of logic-level parallelism in combination with gate up-sizing can be achieved with any standard logic synthesis tool provided that the circuit is not yet subject to the strictest timing constraints.

4.3 Advanced Supply Voltage Scaling

4.3.1 Adaptive Supply Voltage Scaling

The design of electronic circuits is often subject to peak performance constraints. Signal processing units, for instance, are designed to deliver a certain maximum throughput at a given clock frequency and a given supply voltage. When such a module is built into different applications that require different throughputs, the power consumption can be optimized by choosing the clock frequency and the supply voltage in each individual case, such that the actual peak performance constraints are just met. However, if the application does not require maximum performance at all times, such fixed voltage designs result in a waste of energy. The power consumption can then be further optimized by continuously adjusting the clock frequency and the supply voltage according to the instantaneous performance requirements (speed-adaptive supply voltage scaling).

The basic concept of speed-adaptive supply voltage scaling for synchronous circuits defines two main tasks. Firstly, the clock frequency is set according to the instantaneous performance requirements. Secondly, the supply voltage is adjusted such that proper operation of the circuit at the given clock frequency is assured.

Speed-adaptive supply voltage scaling can also be applied to asynchronous modules in a synchronous environment. In this case, of course, no clock frequency scaling is required. Examples of such self-timed system architectures can be found in the literature [86]. Since standard-cell-based ASIC design is always synchronous, only synchronous design concepts are of interest in this study.

In Figure 4.1a, the architecture of a synchronous DSP with speed-adaptive supply voltage scaling is shown [38]. The input data samples are stored in a first-in-first-out (FIFO) input buffer before they reach the DSP core. The buffering increases the latency in the datapath, which is a drawback of this architecture. The workload filter monitors the filling ratio of the FIFO and computes a suitable value for the clock frequency such that FIFO over- and underflow are prevented. The control loop composed of the rate controller, the pulse width modulated (PWM) switching power supply (DC-DC converter), and the voltage controlled oscillator (VCO) is similar to a phase-locked loop (PLL). The rate controller checks whether the clock signal generated by the VCO is in phase with an external reference signal. It also checks whether the clock frequency is equal to the value computed by the workload filter. The output of the rate controller can be interpreted as an error signal. It is used as an input to the duty cycle control logic of the switching power supply and, thus, determines the supply's output voltage. The voltage provided by the power supply is used as a supply voltage for the DSP and as a control voltage for the VCO that generates the clock signal for the DSP. The DSP can be any standard DSP core. The only requirement is that the relation between the worst-case delay in the critical path of the DSP (taking into account variations of process parameters and operating conditions) and the supply voltage must be known. This relation must be properly modeled by the VCO control characteristics.

A different architecture for (speed-)adaptive supply voltage scaling can be seen in Figure 4.1b [66, 104]. In this example, the clock signal and the supply voltage for the RISC (reduced instruction set computer) processor core are not generated within the same control loop. The purpose of the architecture shown in the figure is to automatically adjust the supply voltage such that the delay in the critical path of the processor core matches the period of the given clock signal. For this purpose, the actual delay in the critical path is measured and compared with the clock period. In order to avoid a negative impact of the delay measurement on the operation of the processor, the measurement is carried out on a replica of the critical path implemented in the speed detector. The duty cycle control logic of the PWM switching supply (DC-DC converter) then adjusts the supply voltage for the processor according to the result of the delay measurement.

The oscillator that generates the clock signal is not part of the basic architecture depicted in Figure 4.1b. In fact, this architecture can be used with or without speed-adaptive clock frequency scaling. In the latter case, it is simply called adaptive supply voltage scaling. The

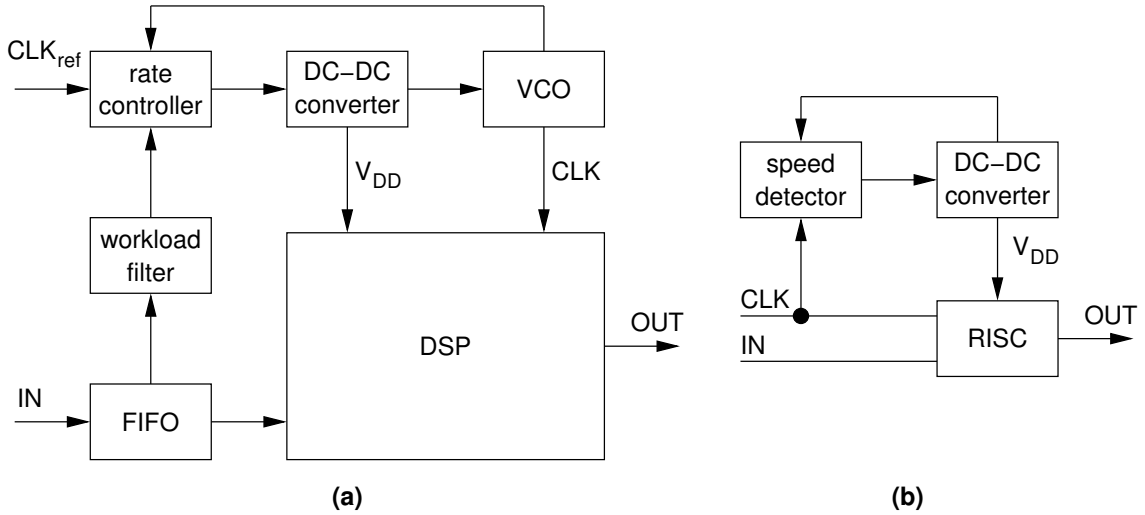

Figure 4.1: Adaptive supply voltage scaling: (a) DSP core with adaptive voltage and frequency regulation [38]; (b) RISC core with adaptive voltage regulation [66, 104].

delay measurement mechanism assures that the supply voltage is adapted not only to any given clock frequency but also to the actual process parameters and operating conditions such as the temperature. This makes the architecture superior to the previous example and to non-adaptive supply schemes, which require the supply voltage to be chosen on the basis of worst-case parameter sets. Significant power savings can be realized in the case of more typical operating conditions and process parameters [2]. If speed-adaptive clock frequency scaling is desired, a suitable hardware or software controlled frequency adaption scheme can be added to this architecture.

Both scenarios discussed above do not impose special requirements or restrictions on the design of the processing units; any state-of-the-art design methodology, including standard-cell-based design styles, can be used. On the other hand, the supply voltage and clock frequency control schemes are difficult to design and usually require tremendous manual design effort.

The concept of (speed-)adaptive supply voltage scaling was first proposed by Kaenel et al. [53]. In recent years, it has been implemented primarily with different types of low power processors. Two case studies, a DSP and a RISC core, have been mentioned above. Other interesting examples are different ARM[5] prototypes [14, 90] and the commercial Crusoe (see [11]) and XScale processors by TRANSMETA[6] and INTEL[7].

[5]http://www.arm.com
[6]http://www.transmeta.com
[7]The former StrongARM architecture is now INTEL'S XScale microarchitecture (http://www.intel.com).

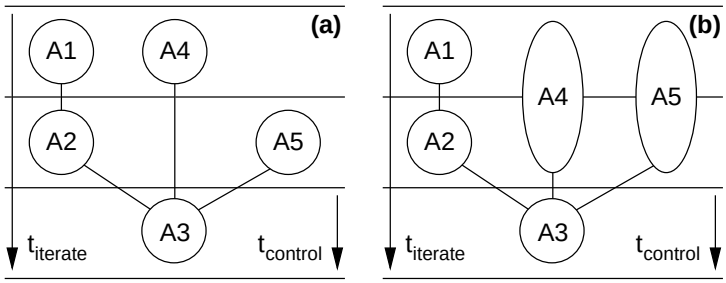

Figure 4.2: A simple data flow scheduled under strict timing constraints [70]: (a) SSV schedule; (b) DSV schedule.

4.3.2 Multiple Supply Voltage Scheduling

Multiple supply voltage (MSV) scheduling is an algorithmic-level power optimization technique. Instead of reducing the critical path delay, as with algorithmic speed-up transformations, MSV scheduling aims at locally scaling down the supply voltage for individual data-path modules that are not on a critical path.

Most MSV scheduling algorithms operate on acyclic DFGs [22, 52, 70, 94]. The DFGs in Figure 4.2 are examples of acyclic graphs representing a simple algorithm that requires five add operations (A1 to A5) to be performed. These graphs are different from the graphs discussed in the context of algorithmic transformations in that there are no loops (iterations) and no weighted edges. However, from any general DFG, a corresponding acyclic DFG can be derived by removing all the weighted edges, and MSV scheduling algorithms can then be used for scheduling the operation within one iteration period ($t_{iterate}$).

In Figure 4.2a, the five operations are assigned control time slots that are represented by the space between two neighboring horizontal lines. These control time slots may be, but do not have to be, identical with actual clock cycles. A DFG with all operations assigned to control time slots is called a schedule.

In the original single supply voltage (SSV) schedule shown in Figure 4.2a, the execution time of every operation matches one control time slot ($t_{control}$) and the critical path (A1, A2, A3) does not contain any slack. However, there is a slack of one time slot in each of the other two paths. Conventional scheduling methods would exploit this slack for resource sharing, i.e. using the same hardware unit for computing A4 and A5, thus reducing the circuit area. Multiple supply voltage scheduling exploits this slack for power optimization by choosing two separate hardware units that both run at a lower supply voltage and, hence, at a lower speed, as indicated in the DSV schedule shown in Figure 4.2b.

The optimization potential increases if the timing constraints are relaxed, i.e. $t_{iterate}$ is increased. If, for instance, the computation of the operations A1 to A5 were allowed to take four control time slots instead of three, the supply voltage could be reduced even for opera-

tions on the critical path, i.e. A1, A2 or A3. Furthermore, a third even lower supply voltage could be used for operations on the paths with the largest slack.

Scheduling under consideration of multiple supply voltages creates delay, power and area overheads that affect the overall effectiveness of this optimization method. The most obvious cause for overheads is additional circuitry such as level converters and multiplexers. Multiple supply voltage scheduling methods can produce realistic and useful results only if the overheads are taken into account. Furthermore, a variety of different data-path modules should be allocated, modeled, and characterized before the scheduling is carried out. However, the complexity of the modules and the large number of parameters affecting their timing and power characteristics can easily make this task unmanageable regarding the computation time and the amount of data that has to be stored. Satisfactory solutions to this high-level macro modeling problem are still lacking, which is one reason why high-level synthesis has not yet made the breakthrough.

Most of the recent work is based on extremely simplifying assumptions. In [94], for instance, the timing and power characteristics of all data-path units are assumed to be identical with the supply voltage being the only parameter. The overheads mentioned before are either completely ignored or considered to be negligible. The published results imply that MSV scheduling subject to the strictest timing constraints using two supply voltages (5 V and 3 V for 0.8 µm CMOS) can reduce the power consumption by 24% on average. This number, however, is highly questionable because of the simplifications.

The results presented in [22] are more useful because the problem was not over-simplified. A selection of different data-path modules was used, including multipliers, adders, and subtracters. Simplifications were made for timing and power modeling, but all modules were characterized on the basis of their actual implementations. This means that the type and the internal structure of the modules was reflected by the model data. Also, the data dependency of the power consumption was included in the power model and considered during the characterization. The overheads caused by level converters and multiplexers were taken into account, but the effect of different scheduling options on the control logic was neglected.

The results clearly show the effect of the more realistic problem formulation. Using three supply voltages (5 V, 3.3 V, and 2.4 V) and assuming the strictest timing constraints, the MSV scheduling method yielded an average power reduction of only 4%. Thus, MSV scheduling does not appear to be effective if the performance is tightly constrained.

4.3.3 Logic-Level Dual Supply Voltage Scaling

The purpose of logic-level DSVS is to reduce the supply voltage for gates in non-critical paths from the nominal value V_{DD} to a lower value V_{DDL}. Figure 4.3 illustrates a typical DSV circuit structure. In DSV circuits, low voltage cells must not directly drive high

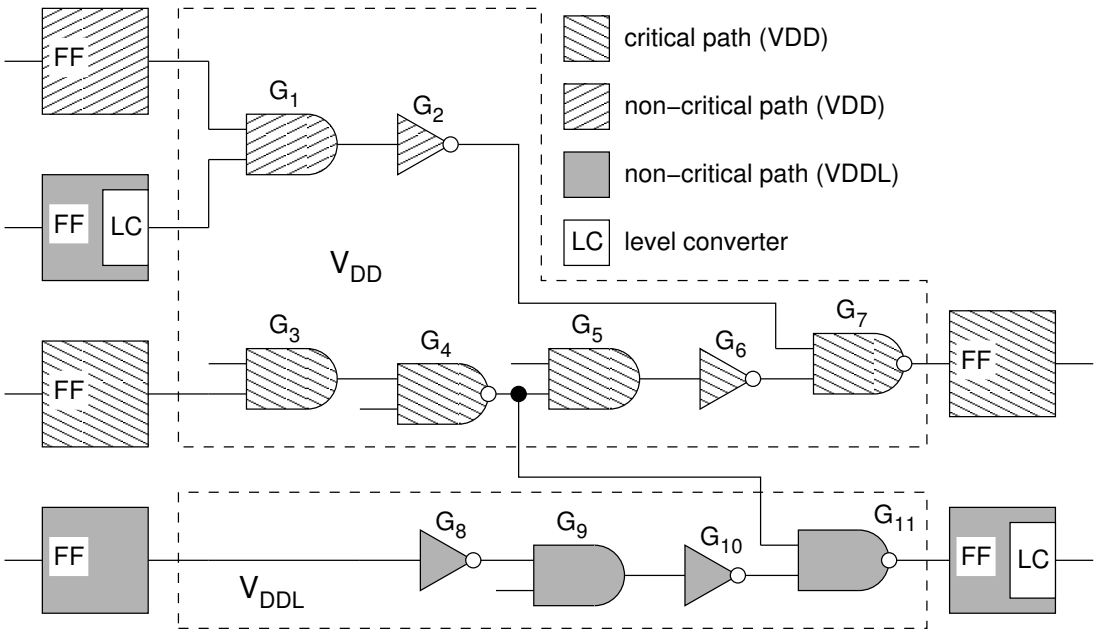

Figure 4.3: DSV circuit structure.

voltage cells. Otherwise, quiescent currents occur at the driven gates. This is why the gates G_1 and G_2 in Figure 4.3 are operated at V_{DD} although they are part of a non-critical path.

Level-converting cells may be inserted where transitions from V_{DDL} to V_{DD} are required. These cells, however, introduce additional delay and cause power and area overheads. The overhead is minimized, if level conversion is enabled only at the input and output nodes of combinational blocks, as depicted in Figure 4.3.

Automated dynamic power optimization using logic-level DSVS was demonstrated by several researchers [20, 113, 122]. Power savings in the range of 10% (see [113]) to 45% (see [20]) were achieved for individual circuit examples. Further details on the optimization technique and on the results of related work follow in Chapter 5.

A shortcoming of the published work in this field is that DSVS was not carried out under realistic conditions. Particularly, the true additional benefit of DSVS methodologies in comparison with state-of-the-art power-driven logic synthesis was not investigated. Nevertheless, since DSVS is suitable for automated dynamic power optimization and does also not impose constraints on the choice of fabrication process, the technique fulfills two important requirements defined in Section 1.3.

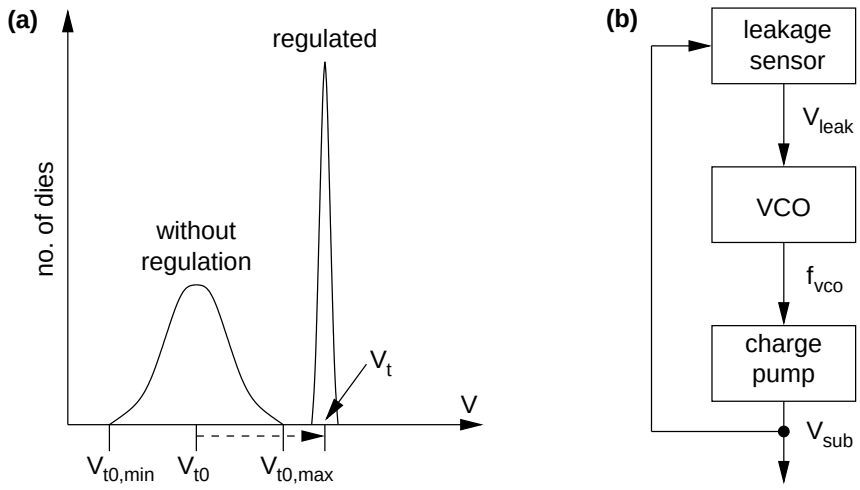

Figure 4.4: Leakage-controlled threshold voltage regulation: (a) threshold voltage distribution with and without regulation [85]; (b) leakage-controlled voltage source [54, 65].

4.4 Advanced Threshold Voltage Scaling

4.4.1 Leakage-Sensitive Threshold Voltage Regulation

The impact of threshold voltage uncertainties on the choice of nominal supply and threshold voltages has been discussed in Section 4.1. The smaller the uncertainties are, the closer to the optimum the nominal voltages can be chosen, which keeps the power consumption close to the minimum. As explained in Section 2.1, the threshold voltage depends on the source-bulk voltage (body effect). The body effect can be exploited for electronic threshold voltage control that minimizes the threshold voltage uncertainties. This requires a technology where the absolute values of the zero-bias threshold voltages V_{t0} of all transistors are below the desired nominal value V_t under all circumstances. More precisely, the desired V_t value must be equal to or larger than the maximum possible zero-bias threshold voltage $V_{t0,max}$, taking into account process and temperature variations as depicted in Figure 4.4a [85]. The reason is that the source-bulk voltage must always be positive for n-channel transistors and negative for p-channel transistors and, hence, the threshold voltage can be shifted in only one direction.

For threshold voltage regulation, the n- and p-wells are connected to leakage-controlled voltage sources such as the one depicted in Figure 4.4b [54, 65]. In this example, the substrate voltage V_{sub} comes from a charge pump. It depends on the frequency f_{vco} of the input signal to the charge pump. This signal is taken from the output of a VCO which is controlled by a leakage sensor. The sensor measures the subthreshold current flowing through a reference transistor and converts it to an equivalent voltage V_{leak}. The reference transistor is operated in the subthreshold region, where the exponential current-voltage

characteristics make the drain current (subthreshold/leakage current) very sensitive to small deviations of the threshold voltage from the desired value. This results in precise regulation.

Kuroda et al. applied this principle to a cosine transform processor [64]. They designed the circuit for a 0.3 µm CMOS process with zero-bias threshold voltages of 0.15 V ± 100 mV. The regulation scheme was used for adjusting the threshold voltage to 0.27 V ± 20 mV. More information on this design can be found in the following section about switched threshold voltages. Other case studies published by Kuroda et al. are an MPEG-4 video codec [116] and a complex MPEG-4 video phone chip [87].

This regulation scheme addresses only the problem of die-to-die threshold voltage variation. Extending the approach to within-die variation would require a larger number of separated control loops which would, of course, increase the area and power overhead.

The implementation of such a threshold voltage regulation scheme is complicated. The circuitry, particularly the leakage sensor, is difficult to design and, just as in the case of adaptive supply voltage scaling, tremendous manual design effort is required. For standard-cell-based design with threshold voltage regulation, the layouts of all cells in the library have to be modified such that the n- and p-wells are disconnected from the power and ground rails. Instead, pins for substrate voltage routing must be assigned to the well contacts.

4.4.2 Switched Threshold Voltages

The threshold voltage regulation scheme described in the previous section can also be exploited for switching between low threshold voltages in active modes and high threshold voltages in inactive modes. This way, sufficient performance can be assured in active modes and subthreshold leakage currents can be reduced in inactive modes.

Kuroda et al. applied this principle to the cosine transform processor that has been mentioned in the previous section [64]. In the active mode, the absolute values of the nominal threshold voltages were set to 0.27 V as mentioned above. This required substrate voltages of -0.5 V for the n-channel and +1.4 V for the p-channel devices. In the standby mode, the absolute values of the nominal threshold voltages were increased to values above 0.5 V, which required substrate voltages of -3.3 V and +4.2 V for the n-channel and p-channel devices, respectively. This technique led to a reduction of the subthreshold currents from 100 µA in the active mode to only 10 nA in the standby mode. The power overhead created by the leakage sensors was 0.1% of the total power consumption in the active mode and 10% in the standby mode. The same principle was applied to the MPEG-4 video codec [111] and the MPEG-4 video phone chip [87] that have been mentioned before.

Although the substrate voltages are larger than the supply voltage of 1.0 V in the above example, the use of a charge pump enabled all voltages to be derived from the output of a single battery cell. A potential problem with the switched threshold voltage approach, however, is that even larger substrate voltages will be needed in future technologies. As

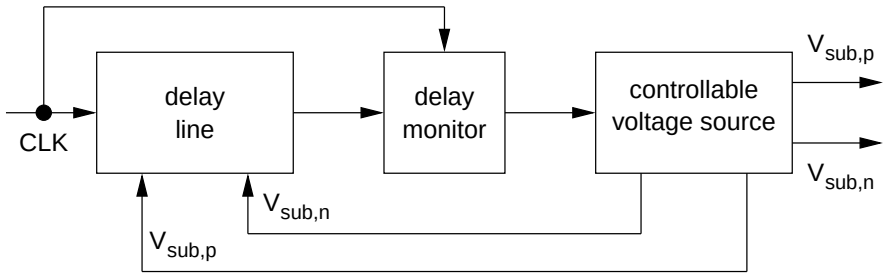

Figure 4.5: Circuit configuration for speed-adaptive threshold voltage scaling [81].

the minimum feature size is scaled down, the nominal threshold voltage values are reduced as well, which requires lower substrate doping concentrations N_A and N_D for n-channel and p-channel transistors (see Equations 2.11 and 2.12), respectively. According to Equation 2.13, lower doping concentrations also reduce the body factor. Consequently, larger substrate voltages are necessary for the switched threshold voltage approach. Some researchers have even forecasted that threshold voltage manipulation through body bias may not be possible in future technologies because of too small body factors [10].

4.4.3 Speed-Adaptive Threshold Voltage Scaling

Another threshold voltage regulation scheme aims at constant delay as opposed to constant threshold voltage or subthreshold current [81]. Just as in the regulation scheme described above, the body effect is exploited for controlling the threshold voltages. The fundamental difference is that the delay is monitored instead of the subthreshold current.

Figure 4.5 illustrates the concept. The delay monitor measures the delay of a delay line with respect to the external clock signal CLK. The delay line is a chain of inverters that receives the external clock signal at its input and provides a delayed clock signal at its output. The output signal of the delay monitor is proportional to the deviation of the actual delay of the delay line from the desired delay. It is used as an input to a controllable voltage source that generates the variable substrate voltages $V_{sub,n}$ and $V_{sub,p}$ for n-channel and p-channel transistors in the delay line and in the functional blocks.

This approach allows delay variations caused by temperature variations, short-channel effects or process parameter deviations to be minimized. Also, the concept of switched threshold voltages can be adopted in order to support different modes of operation. In this sense, the speed-adaptive threshold voltage scaling is equivalent to the leakage-sensitive threshold voltage regulation. An advantage of the speed-adaptive approach is that even delay variations due to supply voltage fluctuations can be compensated. On the other hand, the leakage-sensitive approach can be combined with speed-adaptive supply voltage scaling, which is not possible in the case of speed-adaptive threshold voltage scaling for regulation stability reasons.

4.4.4 Dual Threshold Voltage Techniques

High threshold voltage power switches. At the NTT LSI Laboratories in Japan, a special circuit technique for the suppression of subthreshold currents during periods of inactivity (standby mode) was developed. The circuit technique is usually denoted MTCMOS, although strictly speaking this abbreviation only describes the type of technology required for implementing such circuits, i.e. multiple threshold voltage CMOS technology. Several experimental MTCMOS designs were reported. These include a PLL [83] and several DSPs for mobile phones [84], and other applications [51].

In MTCMOS circuits, the logic gates are implemented with low threshold voltage transistors in order to provide sufficient performance in the active mode. These gates are not connected directly to the primary power and ground rails. Instead, they are connected to virtual power and ground lines, which are connected to the primary rails via high threshold voltage power switch transistors. In the standby mode, the power switches are turned off which practically separates the logic gates from the power supply. The subthreshold currents are low in this mode of operation because of the high threshold voltages of the power switch transistors.

Adding power switch transistors to every cell in a standard-cell-library requires a tremendous design effort and is, therefore, impractical. In most MTCMOS designs, power switch transistors are placed at both ends of each cell row [83, 84]. While this reduces the design effort, it of course requires devices with extremely large channel widths.

One issue in the design of MTCMOS circuits is to preserve the state of sequential cells in the standby mode. This can be accomplished by means of a special type of flip-flop [97]. A more serious problem is that additional transistors are connected in series with the pull-up and pull-down paths of the logic gates [56, 57, 83]. Firstly, this increases the series resistance in the active mode. Secondly, the voltage drop across the power switches causes the threshold voltages of the low threshold voltage transistors to increase due to the body effect. Thirdly, the currents flowing through the power switches change whenever there is activity in the circuit. This results in voltage fluctuations on the virtual power lines with the consequences discussed in Section 4.1. All this degrades the performance and sets limits to low voltage operation.

Dual threshold voltage logic synthesis. The purpose of logic-level dual threshold voltage scaling (DTVS) is to increase the threshold voltage for gates in non-timing-critical paths from the nominal value V_t to a higher value V_{thigh} [119, 103]. The idea is similar to that of DSVS. However, the primary goal is to reduce the static power consumption in both active and inactive modes, rather than the dynamic power as in the case of DSVS.

Dual threshold voltage (DTV) circuits do not require level conversion as indicated in Figure 4.6. Rather, low and high threshold voltage cells can be cascaded arbitrarily. This makes DTVS easier to use than DSVS.

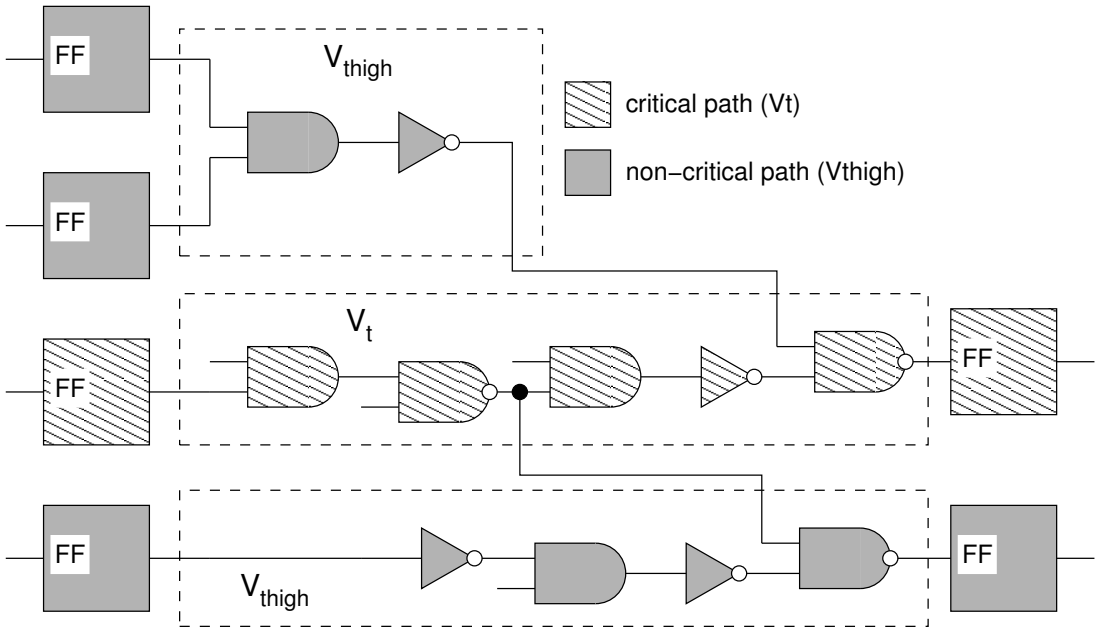

Figure 4.6: DTV circuit structure.

It was demonstrated that the DTVS technique can be carried out with standard tools and that it can, hence, be easily incorporated in state-of-the-art logic synthesis environments [42]. The only additional requirement is the availability of a DTV technology and a DTV library, i.e. a library that contains high and low threshold voltage versions of all cells. Under these conditions, a power-conscious synthesis tool can always choose between a high speed and a low leakage implementation of a certain logic function.

Just recently, design software vendors have begun to officially promote DTV logic synthesis for static power optimization [109]. As a result, this technique is actually on the way to becoming state-of-the-art. The largest obstacle probably is that many foundries and other semiconductor vendors do often not provide access to DTV technologies and libraries.

Although DTVS is primarily meant for static power optimization, it can also be useful if the focus is on the dynamic power consumption. A conceivable strategy would be to scale down the nominal supply and threshold voltages beyond the optimum (see Section 4.1), so as to reduce the dynamic power consumption at the cost of larger subthreshold currents without degrading the performance, and then minimize the static power by using transistors with larger threshold voltages in non-timing critical paths.

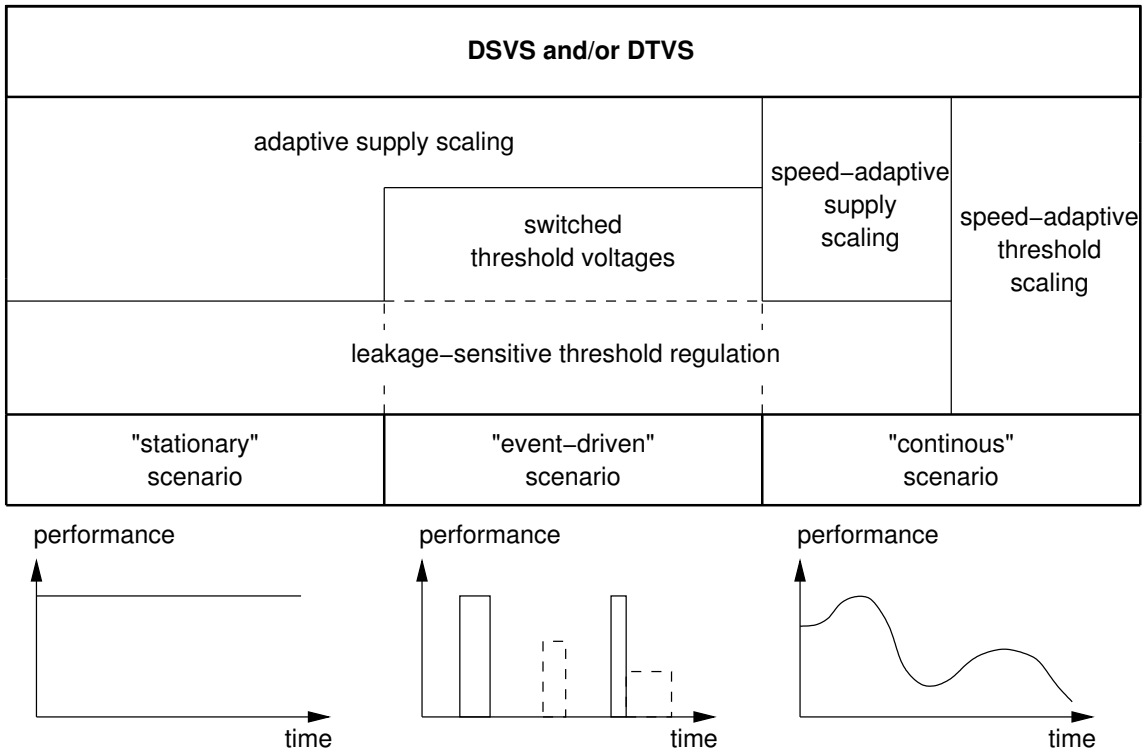

Figure 4.7: Applicability of advanced voltage scaling to different circuit categories.

4.5 Circuit Classification for Advanced Voltage Scaling

Electronic applications can be classified with regard to their demand on the circuit performance. Figure 4.7 shows three typical performance and voltage scaling scenarios [74].

In the ”stationary” scenario, the demand on the performance is invariant with time. This is typical of many image processing applications, where the computational effort depends only on the image size. If the threshold voltage uncertainty is minimized by means of leakage-sensitive threshold voltage regulation, the nominal supply and threshold voltages can be reduced to near-optimal values, which improves the energy efficiency at given performance and leakage specifications. Alternatively, speed-adaptive threshold voltage scaling could be used for coping with voltage uncertainties.

In the ”event-driven” scenario, high performance is required only during short periods of time, which is typical of the classical cellular phones [84]. In such applications, the MTCMOS circuit technique could be applied. However, this does not appear to be feasible in the case of very low supply voltage operation, as explained in Section 4.4.4. The more promising approach is to exploit the body effect for switching threshold voltages between low and high values in active and inactive modes, respectively. As explained before, this can

be combined with both leakage-sensitive threshold voltage regulation and speed-adaptive threshold voltage scaling.

In the ”continuous” scenario, the demands on the circuit performance are varying continuously with time. This is the case, for instance, if the computational effort depends on the image content in image processing. In such applications, leakage-sensitive threshold voltage regulation can be combined with speed-adaptive supply voltage regulation in order to keep the dynamic power at a minimum at all times. Theoretically, speed-adaptive threshold voltage scaling could be used for keeping static power at a minimum while continuously adjusting the circuit delay to the optimum. However, results obtained with such an approach have not been published yet.

The DSVS and DTVS techniques appear to be somewhat more versatile than the others. Both techniques are, in principle, applicable in all three scenarios as depicted in Figure 4.7. They can even be combined with different voltage regulation techniques or with each other. For instance, DSVS was used together with leakage-sensitive threshold voltage regulation, switched threshold voltages, and adaptive supply voltage scaling in an MPEG-4 codec core [40, 67, 111, 116]. A combination of DSVS and DTVS was also proposed [60]. Using DTVS together with adaptive supply voltage scaling appears to be feasible and promising, although no results have been published yet.

4.6 Summary, Comments, and Conclusions

The two high-level strategies discussed in this chapter, i.e. transformation-based algorithmic speed-up and MSV scheduling, are too complex to be used without tool assistance. Since, on the other hand, high-level synthesis is still far away from being state-of-the-art, these techniques are currently of no real practical use in the design of standard-cell-based ASICs, which is not expected to change in the short term.

Architecture-driven supply voltage scaling is also not supported by existing design tools, but the ideas are simple enough to be hand-coded in the RTL modeling phase of an ASIC design process. Thus, architecture-driven supply voltage scaling is considered to be state-of-the-art. However, this study is based on the assumption that the concepts of parallelism and pipelining were exploited appropriately in the development of the designs that are used as test cases and further architectural changes are not made.

Global supply voltage scaling (GSVS) enabled through logic-level parallelization and gate up-sizing is a simple concept that can be realized in state-of-the-art logic synthesis methodologies. Under certain conditions, it can be viewed as an alternative to the methodology proposed in this study. Therefore, GSVS is embraced in the evaluation of the novel methodology developed in this work (see Chapters 7 and 8).

Leakage-sensitive threshold voltage regulation, switched threshold voltages and (speed-) adaptive supply voltage scaling are very promising especially for ultra-low-voltage opera-

tion. An obstacle for using any one of these techniques is the tremendous effort of designing the complex regulation schemes involved (full-custom design of mixed-analog-digital circuits). This makes such techniques an interesting option for the design of standard products like low power general purpose microprocessors. In the case of standard-cell-based ASICs, however, the extra cost and the increased time-to-market are usually not acceptable.

As mentioned before, a major advantage of DSVS and DTVS over the various regulation techniques is versatility, which means these techniques appear to be suitable for a broad range of applications with quite different characteristics. Furthermore, both techniques can be automated as a part of the logic synthesis and are, hence, well suited to standard-cell-based design.

This study focuses on automated dynamic power optimization in standard-cell-based ASIC design. This can be accomplished by means of DSVS or DTVS or both. An important constraint in this work is the availability of fabrication processes and standard-cell-libraries that provide no more than one threshold voltage. Therefore, it was decided to develop a power-driven logic synthesis methodology that incorporates DSVS and allows an investigation of the potential and the limitations of this technique under realistic conditions.

Chapter 5

Logic-Level Dual Supply Voltage Scaling

The purpose of DSVS at the logic level is to selectively scale the supply voltage for gates in non-timing-critical paths from the nominal value V_{DD} down to a lower value V_{DDL} in order to reduce the dynamic power consumption. This is accomplished in two consecutive design phases. First, the DSV circuit structure is generated. Preferably, this is done in the technology dependent phase of logic synthesis after technology mapping (post-mapping optimization). In the subsequent layout synthesis phase, the cells are placed in such a way that each cell can be supplied with the correct voltage. The methodology developed in this work addresses the problem of DSV post-mapping optimization.

The fundamentals of DSV post-mapping optimization, including the circuit structure (see also Section 4.3.3) and the level conversion issue, are explained in this chapter. Timing conditions for the applicability of voltage scaling to individual gates are formulated, and an expression describing the expected power savings is derived. The extension of DSV post-mapping optimization to voltage scaling in the clock network is explained as well. These explanations include an expression describing the additional power savings and the power overheads associated with the use of clock voltage scaling.

The discussion of DSV post-mapping optimization and clock voltage scaling is followed by a review of relevant related work. Finally, existing solutions to the DSV placement problem are discussed, in order to emphasize the general feasibility of the entire concept of DSVS at the logic level.

5.1 Dual Supply Voltage Post-Mapping Optimization

5.1.1 Dual Supply Voltage Circuit Structure

Figure 4.3 illustrates a DSV circuit structure. The circuit has one critical path, where all the gates (G_3 to G_7) are operated at V_{DD} in order to satisfy the timing constraints. Furthermore,

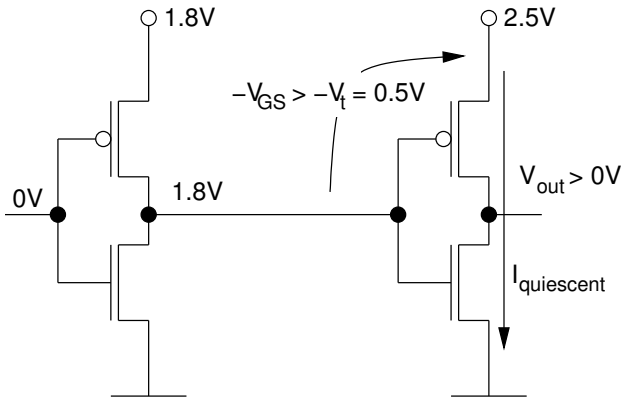

Figure 5.1: Consequences of a low voltage gate driving a high voltage gate [113]. The values are typical of 0.25 µm CMOS technologies.

there is one non-critical path that exhibited enough slack for the supply voltage of all the gates (G_8 to G_{11}) to be scaled down to V_{DDL}. The gates G_1 and G_2 form another non-critical path. Nevertheless, these gates must be operated at V_{DD}, as a consequence of the level-conversion issue explained in the following section.

5.1.2 Level Conversion

As can be seen in Figure 5.1, the p-channel transistors in high voltage gates driven by low voltage gates are always conducting, if the difference of V_{DD} and V_{DDL} is larger than the threshold voltage V_t [113]. Consequently, excessive quiescent currents occur at the driven gates. In the worst case, the outputs of the high voltage gates may be invalid resulting in failure of operation. Consequently, low voltage cells must not directly drive high voltage cells, and the gates G_1 and G_2 in Figure 4.3 must be operated at V_{DD} although being part of a non-critical path.

The different voltage levels do not cause any problem in the case of high voltage gates driving low voltage gates; an input signal level higher than the supply voltage is always sufficient to turn off the p-channel transistors in the driven gate. Therefore, the output of the low voltage gate G_4 is connected directly to the input of the high voltage gate G_{11} in the circuit shown in Figure 4.3.

Level-converting cells could be inserted where transitions from V_{DDL} to V_{DD} are required. However, these cells introduce additional delay and cause power and area overheads. These overheads can be minimized by enabling level conversion only at the input and output nodes of combinational blocks, which has two advantages. Firstly, the total number of level converter cells is minimized. Secondly, the remaining level converters can be merged with flip-flops, as depicted in Figure 4.3, which further reduces the overhead [116].

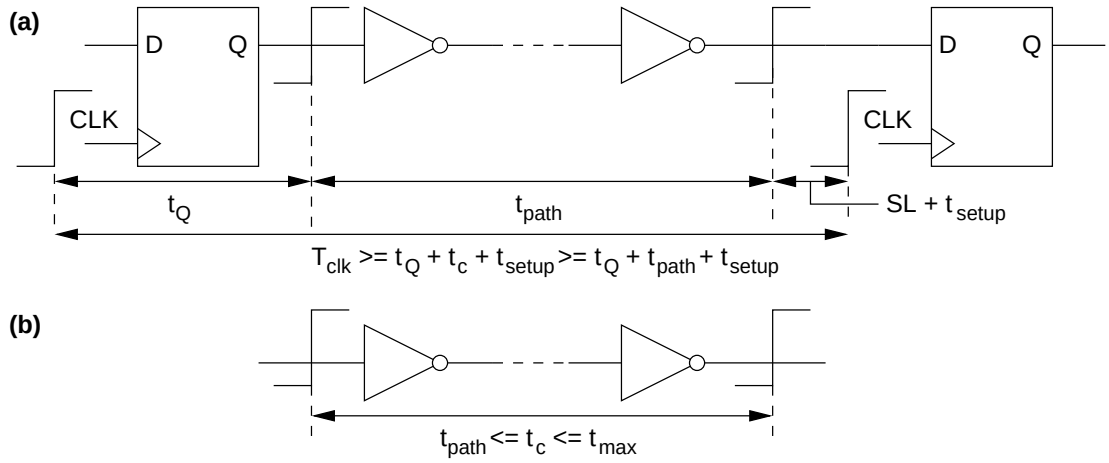

Figure 5.2: Timing constraints: (a) synchronous circuit; (b) combinational circuit.

5.1.3 Timing Conditions for the Applicability of Voltage Scaling

Developing timing conditions that determine whether voltage scaling can be applied to individual gates requires a review of the fundamental timing parameters that apply to the logic synthesis of sequential and combinational circuits.

In synchronous digital circuits, the target clock period T_{clk} is an important timing constraint. The sum of the clock-to-output delay t_Q of the register driving the critical path, the critical path delay t_c, and the setup time t_{setup} of the register at the end of the critical path must not exceed T_{clk}. This constraint can be written as

$$T_{clk} \geq t_Q + t_c + t_{setup} \quad . \tag{5.1}$$

The slack of a (not necessarily critical) path with a delay $t_{path} \leq t_c$ is the maximum additional delay that may be introduced into the path without violating the timing constraint given by Equation 5.1. The slack SL is defined as

$$SL = T_{clk} - t_Q - t_{path} - t_{setup} \quad . \tag{5.2}$$

For a purely combinational circuit, the timing constraint is specified as the largest acceptable path delay t_{max}. Thus, the constraint can be expressed as

$$t_{max} \geq t_c \quad , \tag{5.3}$$

and the slack is defined as

$$SL = t_{max} - t_{path} \quad . \tag{5.4}$$

The constraints for sequential and combinational circuits are illustrated in Figure 5.2.

Reducing the supply voltage increases the gate delay (see Equation 2.15). Thus, a necessary condition for voltage scaling being applicable to a gate G_i in a technology-mapped logic-level implementation of a circuit is that the timing slack SL_i of the longest path running through G_i must be larger than the additional delay created by the voltage reduction.

Because of the level conversion issue, the above condition is not sufficient. In fact, three conditions must be fulfilled. Firstly, the slack SL_i must be large enough so that the gate G_i and all the gates along the longest path in the fan-out of G_i can be operated at V_{DDL} at the same time. Secondly, the first condition must be fulfilled for G_i and for all the gates in all fan-out paths of G_i. Thirdly, all flip-flops in the fan-out of G_i must be replaceable with their low voltage or level-converting counterparts without violating the timing constraint defined by Equation 5.1 because of increased setup times or clock-to-output delays.

These conditions form the basis of the power savings estimation method that is developed in Section 6.3.

5.1.4 Expected Power Reduction

In this section, an expression describing the impact of DSVS on the total dynamic power consumption P_{dyn} of a technology-mapped logic-level design is derived. In all equations following in this section, P_{cap}, P_{sc}, and P_{dyn} denote the capacitive power, the short-circuit power, and the total dynamic power, respectively, at a supply voltage of V_{DD}.

The dependence of the capacitive power P_{cap} and the supply voltage V_{DD} is quadratic (see Equation 2.19). The absolute value of the relative capacitive power reduction $\Delta P_{cap}/P_{cap}$ that can be expected from scaling the supply voltage of a circuit from V_{DD} down to V_{DDL} is, therefore, given by

$$\left|\frac{\Delta P_{cap}}{P_{cap}}\right| = 1 - \left(\frac{V_{DDL}}{V_{DD}}\right)^2 \quad . \tag{5.5}$$

The effect of the supply voltage on the short-circuit power P_{sc} is more difficult to predict. On the one hand, Equation 2.22 shows a direct relation of P_{sc} to $V_{DD} - 2V_t$, which implies that P_{sc} decreases with decreasing V_{DD}. On the other hand, P_{sc} is proportional to the input transition time t_T. Since lower supply voltages result in larger output transition times t_{TO} and the output transition time of a gate is identical with the input transition time of the driven gates, reducing the supply voltage for one gate increases the short-circuit power in the driven gates.

A common first-order approximation of the output transition time is $t_{TO} = 2t_D$ [120]. Using Equation 2.15, the aforementioned approximation of t_{TO} results in

$$t_{TO} = \frac{C_{node} V_{DD}}{\beta K_{ISAT} (V_{DD} - V_t)^{\alpha}} \quad , \tag{5.6}$$

which finally leads to the following relation of the short-circuit power to the supply voltage:

$$P_{sc} \propto (V_{DD} - 2V_t)^{\alpha+1} \cdot \frac{V_{DD}}{(V_{DD} - V_t)^{\alpha}} \tag{5.7}$$

From Equation 5.7, the absolute value of the relative short-circuit power reduction $\Delta P_{sc}/P_{sc}$ to expect from voltage scaling can be derived:

$$\left|\frac{\Delta P_{sc}}{P_{sc}}\right| = 1 - \frac{V_{DDL}}{V_{DD}} \left(\frac{V_{DD} - V_t}{V_{DDL} - V_t}\right)^{\alpha} \left(\frac{V_{DDL} - 2V_t}{V_{DD} - 2V_t}\right)^{\alpha+1} \tag{5.8}$$

With Equations 5.5 and 5.8, the relative reduction of the total dynamic power consumption $\Delta P_{cap}/P_{cap}$ that can be achieved by scaling the supply voltage from V_{DD} down to V_{DDL} can be calculated as follows:

$$\begin{aligned} \left|\frac{\Delta P_{dyn}}{P_{dyn}}\right| &= \frac{P_{cap}}{P_{dyn}} \cdot \left|\frac{\Delta P_{cap}}{P_{cap}}\right| + \frac{P_{sc}}{P_{dyn}} \cdot \left|\frac{\Delta P_{sc}}{P_{sc}}\right| \\ &= 1 - \frac{P_{cap}}{P_{dyn}} \left(\frac{V_{DDL}}{V_{DD}}\right)^2 - \frac{P_{sc}}{P_{dyn}} \left[\frac{V_{DDL}}{V_{DD}} \left(\frac{V_{DD} - V_t}{V_{DDL} - V_t}\right)^{\alpha} \left(\frac{V_{DDL} - 2V_t}{V_{DD} - 2V_t}\right)^{\alpha+1}\right] \end{aligned} \tag{5.9}$$

Least-squares curve fitting using the MATLAB[1] software has shown that Equation 5.9 can be approximated by

$$\left|\frac{\Delta P_{dyn}}{P_{dyn}}\right| = 1 - \left(\frac{V_{DDL}}{V_{DD}}\right)^{2+psc} \geq 1 - \left(\frac{V_{DDL}}{V_{DD}}\right)^2 \quad , \tag{5.10}$$

where *psc* is, for instance, 0, 0.14, 0.28, or 0.43 assuming that the short-circuit power accounts for 0%, 10%, 20%, or 30%, respectively, of the total dynamic power (0.25 µm CMOS technology with $V_{DD} = 2.5\,\text{V}$, $V_t = 0.5\,\text{V}$, $\alpha = 1.5$, and $1.5\,\text{V} \leq V_{DDL} \leq 2.5\,\text{V}$).

This shows that the total dynamic power decreases at a slightly larger rate than the capacitive power as the supply voltage is scaled down. Thus, the common practice of estimating the expected dynamic power reduction due to supply voltage scaling considering only the capacitive component, i.e. assuming that *psc* is equal to zero, typically results in a slight underestimation of the actual power savings. These arguments are supported by measured and calculated data presented in [4]. In this study, a conservative estimation of the expected power savings is usually desired and acceptable. Therefore, Equation 5.10 is generally used with *psc* set to zero.

Regarding the analysis of DSV circuits, Equation 5.10 applies only to the group of gates that can be operated at the lower voltage. In order to properly describe the power savings that can be achieved using DSVS on a circuit which is composed of N gates, a parameter ω_i

[1] http://www.mathworks.com

that describes whether voltage scaling is applicable to the *i*-th gate shall be introduced. Equation 5.10 can then be rewritten as follows:

$$\left|\frac{\Delta P_{dyn}}{P_{dyn}}\right| = \left[1 - \left(\frac{V_{DDL}}{V_{DD}}\right)^{2+psc}\right] \cdot \sum_{i=1}^{N} \omega_i \frac{P_{vdd,i}}{P_{dyn}} \geq \left[1 - \left(\frac{V_{DDL}}{V_{DD}}\right)^{2}\right] \cdot \sum_{i=1}^{N} \omega_i \frac{P_{vdd,i}}{P_{dyn}} \quad (5.11)$$

In Equation 5.11, P_{dyn} is the power consumption of the entire circuit before voltage scaling, $P_{vdd,i}$ is the dynamic power consumption of the *i*-th gate when operated at V_{DD}, and ω_i is one if the *i*-th gate can be operated at V_{DDL}, and zero otherwise.

Together with the timing conditions developed in the preceding section, Equation 5.11 forms the basis of the power savings estimation method that is developed in Section 6.3.

5.2 Clock Voltage Scaling

The clock network often accounts for a large portion of the total power consumption of a design, as explained in Section 3.4. Thus, significantly larger power savings can be expected if DSVS is combined with clock voltage scaling.

Reducing the voltage in the clock network from V_{DD} to V_{DDL} requires all registers in the design to be suitable for low voltage clock input signals. Consequently, only low voltage and level-converting flip-flops may be used, as depicted in Figure 5.3.

Level-converting flip-flops typically have inferior timing characteristics, i.e. larger clock-to-output delay and larger setup times, as shown in Section 6.5.2. Therefore, clock voltage scaling always degrades the performance of circuits that are subject to the strictest timing constraints. The shorter the critical path delay, the more severe is the performance degradation. Circuits that are not subject to the strictest timing constraints may still meet the original constraints after reducing the clock voltage. The extra delay that the level-converting flip-flops add to all paths, however, may necessitate more logic-level parallelism or gate up-sizing or both, which causes area and power overheads that detract from the overall effectiveness of clock voltage scaling.

Level-converting flip-flops sometimes consume more power than their high voltage counterparts, which leads to some power overhead. The overhead, in turn, detracts from the power savings obtained through clock voltage scaling. In other cases, level-converting flip-flops may even be more power efficient than the conventional cells, which leads to additional power savings. The power characteristics of level-converting and conventional flip-flops are compared in Section 6.5.2.

Considering the impact of clock voltage scaling on the dynamic power consumption in the clock network, in the combinational parts of the circuit, and in the sequential parts of the

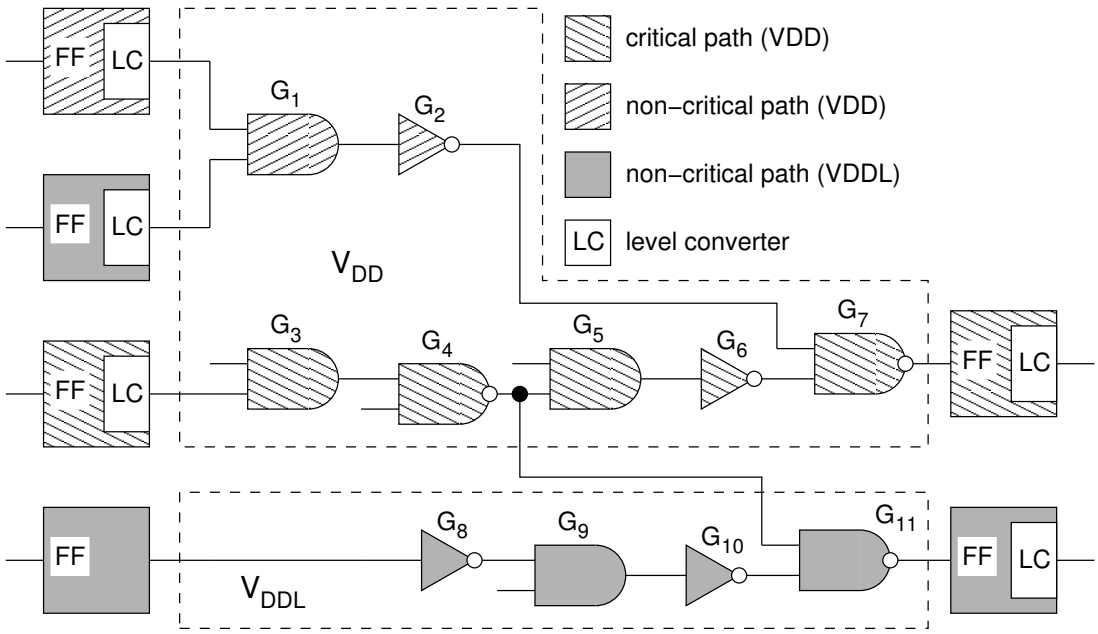

Figure 5.3: DSV circuit structure prepared for clock voltage scaling.

circuit, the overall impact of clock voltage scaling on the total dynamic power consumption can be written as

$$\frac{\Delta P_{dyn}}{P_{dyn}} = \left(\frac{\Delta P_{clk}}{P_{clk}}\right) \cdot \left(\frac{P_{clk}}{P_{dyn}}\right) + \left(\frac{\Delta P_{comb}}{P_{comb}}\right) \cdot \left(\frac{P_{comb}}{P_{dyn}}\right) + \left(\frac{\Delta P_{seq}}{P_{seq}}\right) \cdot \left(\frac{P_{seq}}{P_{dyn}}\right) \quad . \quad (5.12)$$

The first term on the right-hand side of Equation 5.12 describes the primary power savings due to the voltage scaling in the clock network. The second and third terms represent the changes in the power consumption of the combinational and sequential parts of the circuit, respectively. Each of the three terms is a product of two fractions.

The first fraction describes the change in the respective component of power due to clock voltage scaling in relation to the magnitude of that component before scaling the clock voltage. This term can be positive or negative, thus describing a power overhead or a power reduction, respectively.

The second fraction describes the significance of the respective component of power before clock voltage scaling in relation to the total dynamic power of the circuit before scaling the clock voltage. The larger the second fraction in any of the three terms, the more significant is a positive or negative change in the respective component of power. This becomes very clear in the discussion of experimental results in Section 7.6.2.

The clock network itself shows a larger total delay (latency) after voltage scaling. This delay overhead can be reduced by clock buffer sizing if necessary. In this case, some

additional power overhead may result from increased clock buffer sizes. The clock skew, which is the more critical parameter, is hardly affected by voltage scaling and can always be minimized by means of careful clock tree balancing [47]. These aspects can only be analyzed in the post-layout design phase.

For clock voltage scaling to be feasible, firstly, the performance penalty must be acceptable and, secondly, the power that is attributed to the clock network must be a large enough portion of the total power consumption of the design, so that the power overheads can be neglected.

5.3 Related Work

Usami et al. developed a dedicated algorithm for performing clustered voltage scaling (CVS). With the CVS algorithm, the combinational part of a sequential design is split into two partitions, a high voltage and a low voltage gate cluster. Level converters along combinational logic paths are not allowed.

In their early work, Usami et al. applied the CVS algorithm to two selected random logic submodules of the Torch microprocessor using a 0.8 µm CMOS standard cell library [113]. The initial timing-driven synthesis was carried out with SYNOPSYS' DESIGN COMPILER. The same tool was used for timing- and area-driven gate re-sizing immediately after the timing-driven synthesis. State-of-the-art power-driven logic synthesis was not part of the methodology developed by Usami et al.

The nominal supply voltage V_{DD} was fixed at 5 V, while different values were tried for the lower supply voltage V_{DDL}. The optimal choices of V_{DDL} for the first and the second module turned out to be 4 V and 3 V, respectively, and the corresponding power reductions were 10% and 18%. The percentage of low voltage cells in the first and the second module was 48% and 23%, respectively.

The largest acceptable path delay t_{max} was 10 ns, but it is not clear how strict this constraint was. In other words, nothing is said about the shortest possible delay of these two modules. However, a path-based slack distribution analysis performed after gate re-sizing revealed that the second module still contained a relatively large number of non-critical paths, while the number of critical paths was small. This made the power reduction of 18% possible.

In their more recent work, Usami et al. extended the CVS technique to ECVS (Extended CVS), where level converters were allowed to be used along combinational logic paths. Furthermore, clock voltage scaling was applied to clock domains where this could be accomplished without sacrificing the overall performance of the design.

This concept was used in the design of the Mpact media processor in a 0.3 µm CMOS technology with a nominal supply voltage V_{DD} of 3.3 V and the clock frequency set to 75 MHz

[47, 114, 115]. The optimal value for the lower supply voltage V_{DDL} was determined to be 1.9 V.

The ECVS technique was applied to seven random logic submodules of the media processor. In these modules, finally, 8% to 20% of all cells were level converters, which created a power overhead of 8%. Since an average of 76% percent of all cells were operated at V_{DDL}, however, an average power reduction of 28% was achieved in the combinational logic in spite of the overhead.

The power in the entire clock network including all buffers and flip-flops was reduced by 70%. The clock skew was the same as in the original design, which was achieved through careful clock buffer sizing. The clock delay increased by 40%, which did not affect the performance of this design.

The average area overhead of the modules that were optimized using the DSV approach was 15%. Regarding the size of the complete chip, an area overhead of 7% was measured. This overhead was due to placement constraints in the row-by-row layout scheme (see Section 5.4), the large number of additional level-converting cells and the area required for routing the second supply voltage.

As in the first example, i.e. the Torch microprocessor, it is not clear how critical the timing constraints actually were. However, the published path-based slack distribution analysis shows that the Mpact design was obviously even less critical than the two Torch processor modules. About 95% of all paths exhibited a slack of more than 40% of the clock period.

In a third design, an MPEG-4[2] codec core realized in 0.3 µm CMOS, the DSV design concept with the nominal V_{DD} and the optimized V_{DDL} equal to 2.5 V and 1.75 V, respectively, yielded 35% power reduction in the combinational logic and 49% in the clock network [116]. According to the path-based slack distribution analysis, the strictness of the timing constraints was comparable to the Mpact example.

Yeh et al. developed another dedicated DSVS algorithm which they named Gscale and which is basically an improvement of the CVS method [122]. As with CVS, level converters were not allowed to be inserted into combinational logic paths. The major difference between Gscale and CVS is that Gscale uses gate up-sizing in order to increase the slack that can be exploited by DSVS (see Section 6.1).

The algorithm was applied to combinational MCNC benchmark circuits subject to relaxed delay constraints. The circuits were mapped to a 0.6 µm CMOS standard cell library using the experimental SIS[3] software package. First of all, the shortest possible delay of each individual circuit at the nominal supply voltage V_{DD} equal to 5 V was determined. Subsequently, the circuits were re-mapped with the timing constraints, i.e. the largest acceptable path delays t_{max}, set to 1.2 times the shortest possible delays. Unfortunately, the authors did

[2] MPEG-4 is a video coding standard defined by the Moving Picture Experts Group.

[3] SIS is a System for Sequential Circuit Synthesis (free software available from the Department of Electrical Engineering and Computer Science, Electronics Research Laboratory, University of California, Berkely).

not publish the results of a slack distribution analysis. State-of-the-art power-driven logic synthesis was also not part of the methodology developed by Yeh et al.

The Gscale algorithm was then applied to the timing-optimized SSV implementations of the benchmark circuits with V_{DDL} set to 4.2 V. On average, the power was reduced by 19%. The CVS technique yielded 10% power reduction when used on the same circuits.

Chen et al. used yet another dedicated algorithm for optimizing combinational MCNC benchmark circuits subject to varying delay constraints [19, 20]. The algorithm, which is named DVPO (Dual Voltage Power Optimization), allows level converters to be used along combinational logic paths (see Section 6.1).

Chen et al. assumed a technology from the 1 µm generation with a nominal supply voltage V_{DD} of 5 V and a threshold voltage of 0.6 V. The benchmark circuits were first optimized using the SIS package in the minimum delay mode. According to [122], the minimum delay mode results in the fastest possible implementation without regard to the circuit area. Again, state-of-the-art power-driven logic synthesis was not part of this methodology.

The results of a slack distribution analysis revealed that, even though the timing constraints were the strictest, the circuits under consideration contained many non-critical gates and a relatively small number of critical gates.

Each of these circuits was then optimized with the DVPO algorithm under the strictest timing constraints. In other words, the performance of the DSV implementations of the benchmark circuits had to be the same as that of the fastest non-power-optimized implementations produced by SIS. This was done for various different values of V_{DDL}. These experiments revealed that the optimal choice of V_{DDL} can be quite different for individual circuits. For V_{DDL} equal to 3.5 V, on average, 66% of all gates were operated at the lower voltage and the average power reduction was 20%. In additional experiments, even larger power reductions were achieved for selected circuits as the timing constraints were relaxed.

Finally, from a comparison between gate sizing and DSVS, Chen at al. concluded that DSVS is generally more effective than gate sizing. However, in the experiments underlying this comparison, gate sizing led to rather small improvements in power consumption of only 7% on average. This was due to unrealistic assumptions regarding the number and the spacing of cell sizes available in the library [19]. In state-of-the-art logic synthesis methodologies, conventional SSV power optimization leads to larger power reduction, as shown in Section 7.5.1, leaving less room for further improvement through DSVS.

5.4 Layout Synthesis

The second challenge, besides DSV post-mapping optimization in the logic synthesis, are the distribution of two supply voltages across the chip and the layout synthesis. Several solutions to these problems have been published recently.

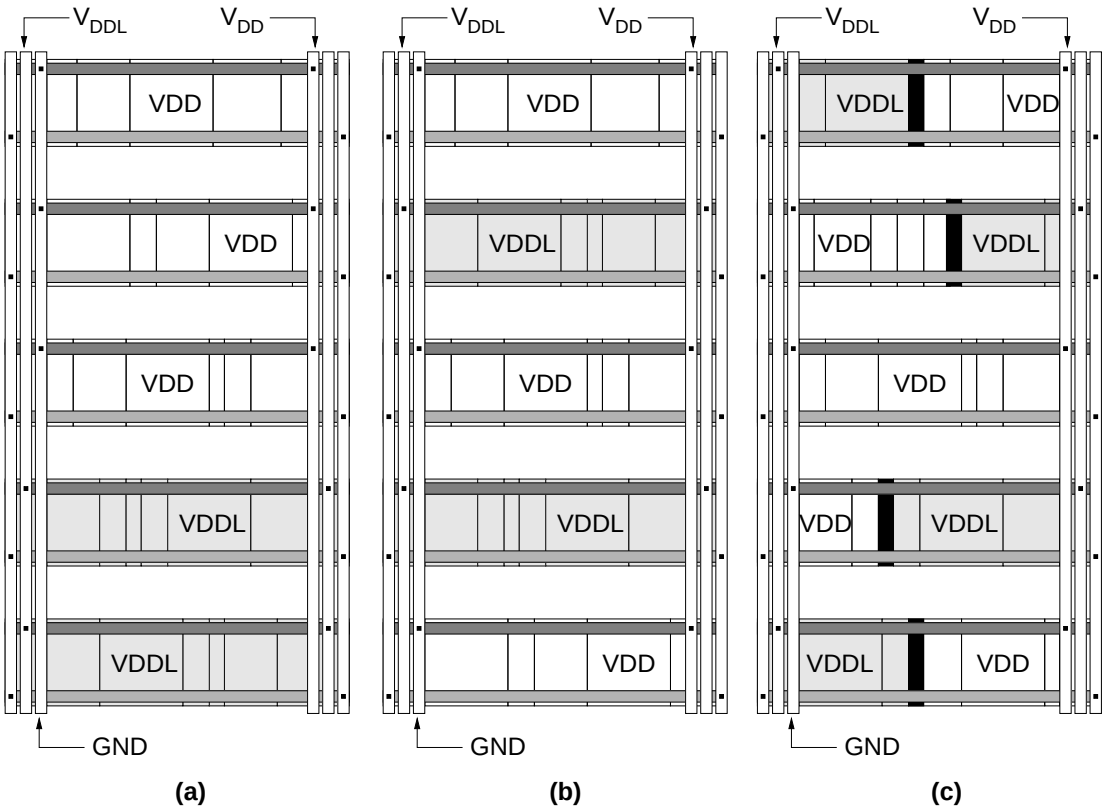

Figure 5.4: DSV layout: (a) macro block style; (b) row-by-row style; (c) split-row style.

Yeh et al. compared four different DSV layout schemes [118, 123, 125]. The three examples shown in Figure 5.4 all aim at voltage separation. In the first case, all cells operated at the same supply voltage are grouped together to form a macro block (see Figure 5.4a). Within each macro block, cells can be placed using existing standard tools but there is significant delay, power and area overhead due to excessive inter-block routing. With the row-by-row layout style (see Figure 5.4b), which was also used by Usami et al. [115], the overhead can be reduced. If the number of low voltage cells is small compared with the number of high voltage cells, however, the number of separate low voltage rows is small and a relatively large interconnect overhead must still be expected. Under these circumstances, the split-row approach (see Figure 5.4c) appears to be preferable. In this third example, the routing overhead is further reduced by splitting each row into a low and a high voltage region separated by special voltage stop cells.

Both the row-by-row and split-row layout schemes require placement tools that are capable of distinguishing low and high voltage cells in order to place them in separate rows or in separate segments within the same row. In the latter case, the tools must also be capable of placing voltage stop cells between low and high voltage segments. While commercial placement tools were not suitable for generating this type of DSV circuit layout in the past,

some widely-used tools now provide means of generating row-by-row or even split-row DSV layouts semi-automatically, although this is not officially supported and results have not been published yet. One commercial placement tool called LAYPAR (available from CATENA[4]) explicitly supports the split-row layout style.

A fourth approach to DSV circuit layout synthesis incorporates modified standard cell layouts as illustrated in Figure 5.5a. Voltage separation is given up and both supply voltages are fed all the way through each row by means of two parallel power rails, so that low and high voltage cells can be placed anywhere in any row. The advantage of this approach is clearly that the automatic placement of the cells does not impose any special requirements on the capabilities of the placement tool. Major disadvantages are a larger chip area and a larger total wire length in consequence of the additional area required for realizing the second power rail in each cell.

Another drawback of the dual power rail scheme is that the cells can be placed directly adjacent to one another only if all n-well regions are connected to the higher supply voltage. Pulling the n-well of low voltage cells to a higher supply voltage (see Figure 5.5b), however, increases the threshold voltage of the p-channel transistors because of the body effect explained in Section 2.1. In a typical 0.25 µm CMOS technology, the worst-case delay of a cell operated at 1.8 V typically increases by about 30% to 50% if the n-well is connected to 2.5 V, as shown in Section 7.5.6. This amount of extra delay must be expected to reduce the number of low voltage cells, thus detracting from the possible power savings. On the other hand, using different bulk potentials for p-channel transistors in low and high voltage cells does not appear to be feasible, because a significantly larger circuit area must be expected due to the extra space required between low and high voltage cells.

Yeh et al. compared the quality of 14 DSV benchmark circuit layouts exploiting split rows on the one hand and dual power rails on the other hand with the quality of conventional SSV circuit layouts. For the split-row layout, about 9% larger area and 11% longer wire length were observed on average [123]. With dual power rail cell layouts the average area and interconnect overheads were 20% and 8% [123]. This clearly shows that the larger cell area causes the circuit area to increase significantly. The interconnect overhead, however, is smaller because low and high voltage cells can be mixed arbitrarily within rows. Usami et al. claim to have realized row-by-row layouts with area overheads as low as 5% [116].

In the work by Yeh et al., pre-layout power analysis yielded an average power reduction of 25% due to DSVS [124]. The actual average power savings after placement and routing were 22% and 20% for the dual power rail and split-row scenarios, respectively [123]. According to these numbers, the dual power rail scheme appears to create less overhead, i.e. 3% as opposed to 5% of the power consumption of a corresponding SSV implementation. However, it seems that Yeh et al. did not consider the performance degradation of low voltage cells due to the increased bulk potential. The synthesis results discussed in Section 7.5.6 indicate that this effect can eliminate the advantage of the better layout

[4] http://www.catena-ffo.de

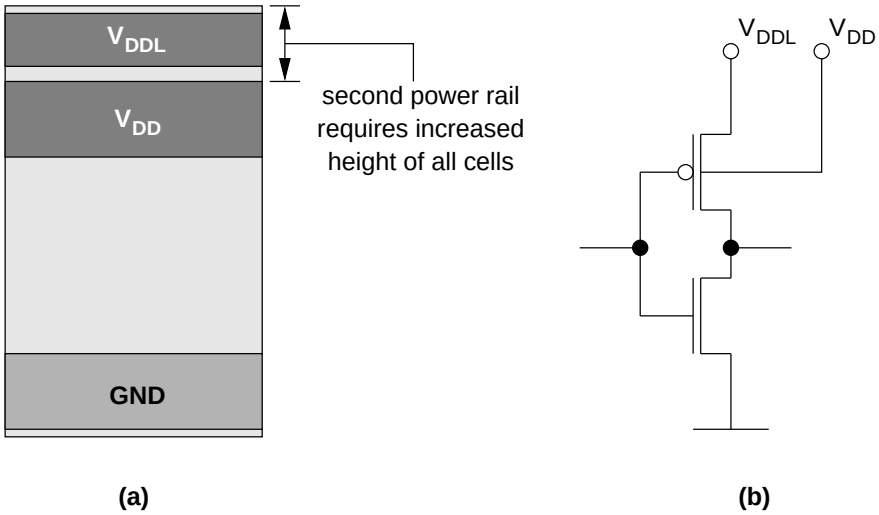

Figure 5.5: Dual power rail standard cells: (a) DSV cell layout with two parallel power rails; (b) low voltage (V_{DDL}) cell with high voltage (V_{DD}) n-well.

quality. Therefore, it appears to be realistic to expect a layout-related power overhead of about 5% of a corresponding SSV implementation or 20% of the power savings achieved in the pre-layout design phase, as observed for the split-row scenario. Unfortunately, the results published by Yeh at al. do not show whether the overhead in the split-row scenario is actually correlated with the number of low voltage cells and, hence, with the power savings achieved in the pre-layout design phase.

Chapter 6

Dual Supply Voltage Logic Synthesis Methodology

In this chapter, a novel power-driven logic synthesis methodology comprising dual supply voltage scaling (DSVS) is presented. This methodology is referred to as the DSV logic synthesis methodology in the remainder of this document.

Besides DSVS, the methodology includes those power optimization techniques that have been identified as being relevant regarding an evaluation of DSVS under realistic conditions (see Chapters 3 and 4). Particularly important in this respect are the technology dependent logic-level techniques that are carried out during state-of-the-art power-driven logic synthesis and, therefore, directly compete with DSVS. In the novel methodology, DSVS and the other logic-level techniques can be used simultaneously. This is crucial for a realistic evaluation of the potential and the limitations of DSVS. Another advantage of this methodology is that it can easily be applied to any type of circuit in standard design environments.

In this chapter, previously published algorithms tailored to DSVS are discussed first. Afterwards, the idea behind the novel methodology is explained. A simple power savings estimation method, that is useful for an analysis of the optimization potential, is developed, as well. Finally, the design and the modeling of a DSV standard cell library, which is the key to the proposed methodology, is discussed.

6.1 Dedicated DSVS Algorithms Used in Related Work

The CVS algorithm by Usami et al. The clustered voltage scaling (CVS) method is based on the depth-first-search concept [113]. The basic structure of the algorithm written in pseudo-code can be seen in Figure 6.1. When the top-level `CVS` procedure is called, the optimization starts with a list of all cells that drive the primary output ports of the gate-level netlist. This list is sorted with regard to the slack of the longest paths running through

```
procedure CVS(netlist) {
  cell_list = cells driving primary outputs of netlist
  sort cell_list so that large slack comes first
  process_cells(cell_list)
}

procedure process_cells(cell_list) {
  foreach cell in cell_list {
    if VDD can be scaled for cell and its fan-out cells {
      scale VDD for cell and all its fan-out cells
      cell_list = cells driving inputs of cell
      if cell_list not empty {
        sort cell_list so that large slack comes first
        process_cells(cell_list)
      }
    }/* end if VDD ... */
  }/* end foreach */
}/* end procedure */
```

Figure 6.1: Pseudo-code for the clustered voltage scaling (CVS) algorithm.

the cells and is then passed to the `process_cells` procedure. Within this procedure, all cells in the list that may be supplied with the lower supply voltage V_{DDL} instead of the nominal voltage V_{DD} are identified. Since level-converting cells are not allowed to be used along combinational logic paths, a cell may only be supplied with V_{DDL} if the cell under consideration and all cells in its fan-out paths can be operated at V_{DDL} without violating the timing constraints. If voltage scaling is applicable to a particular cell, it is actually carried out. After that, another list is created which contains all the cells that drive the input pins of the cell that has just been assigned the lower voltage. This list is then passed to the `process_cells` procedure for recursive processing of the cells in the fan-in paths of the new low voltage cell. The algorithm stops when the last cell from the list of cells driving the primary outputs and all its fan-in cells have been processed. The CVS method performs DSVS on an existing gate-level netlist without performing any other logic-/gate-level transformations at the same time. It was originally implemented as a proprietary re-synthesis tool called POWER SLIMMER.

The CVS method is the reference in this work. It has been implemented in the Tool Command Language (TCL) so that it can be executed from the command line interface of a popular logic synthesis tool. The basic structure of the algorithm has been modeled with standard TCL commands while tool-specific TCL commands have been used for netlist manipulation and static timing analysis. The current implementation works only for combinational circuits, which is sufficient for this work.

The Gscale algorithm by Yeh et al. The Gscale algorithm is an improvement of the CVS method [122]. In the first step, Gscale generates an initial solution, i.e. a first clustering of the given netlist, using CVS. In preparation of the second step, a set of gates which is called the time-critical boundary (TCB) is identified. This set contains all high voltage gates that drive at least one low voltage gate. In other words, the TCB marks the boundary between the high and the low voltage clusters. In the second step, gate up-sizing is applied to the TCB in order to create additional slack. It is usually impossible to increase the size of all gates in the TCB at the same time. Thus, Gscale determines a subset of gates such that up-sizing maximizes the total slack created while the area overhead is minimized. This is a maximum-weighted independent set (MWIS) problem with the weight of a gate defined as the slack increment divided by the area penalty. In the third step, the new slack is exploited for voltage scaling by means of another CVS run. This time, CVS does not start from the primary outputs but from the old TCB. The result is a modified clustering of the netlist with a new TCB. Starting from this new solution, the algorithm continues with the second and the third step in an iterative manner until no further improvement can be achieved. Just as CVS, Gscale does not allow level converters to be used anywhere along combinational logic paths. Yeh et al. implemented Gscale as a part of the academic SIS software package. Apparently, this implementation was restricted to the optimization of purely combinational circuits. For results see Sections 5.3 and 7.1.

The DVPO algorithm by Chen et al. The dual voltage power optimization (DVPO) method is based on another MWIS formulation of the DSVS problem [19, 20]. After an initial static timing analysis all gates in the given netlist that may be individually operated at the lower supply voltage V_{DDL} without causing timing violations are identified. Usually, not all of these gates can be operated at V_{DDL} at the same time without violating the timing constraints. Therefore, the supply voltage is reduced from V_{DD} to V_{DDL} only for an appropriate subset (MWIS) of these gates. This subset is chosen in such a way that, firstly, all gates therein can be operated at V_{DDL} simultaneously and, secondly, the sum of their weights is maximized. In this case, the weight of a gate is defined as the power reduction due to voltage scaling divided by the delay penalty. After scaling the voltage, the timing is re-analyzed and a new set of gates that might be individually operated at V_{DDL} is identified. If this set is not empty, the DVPO algorithm continues with the determination of a new MWIS, voltage scaling, timing re-analysis and so forth. Otherwise it stops. The insertion of level converters along combinational logic paths is allowed. The delay and power overheads introduced by level-converters can be included in the calculation of the weight of a gate that requires such a cell at its output. The DVPO algorithm can be modified, so as to enable simultaneous DSVS and gate down-sizing [19]. In this case, every gate must be checked not only for the applicability of voltage scaling but also for the feasibility of gate down-sizing. If both options are possible, the one with the higher weight is automatically selected when the MWIS is determined. The DVPO algorithm was integrated with the SIS synthesis environment and, apparently, it worked only for purely combinational circuits. For results see Sections 5.3 and 7.1.

6.2 Gate Sizing Algorithms and DSV Cell Modeling

All the algorithms discussed in the previous section were developed specifically for DSVS. They perform DSVS alone (CVS) or DSVS combined with gate up-sizing (Gscale) or DSVS combined with gate down-sizing (DVPO). The structure of the gate-level netlist is not modified in any case. The algorithms were originally implemented as proprietary tools or integrated into academic tools like SIS. One of the main reasons why DSVS has not yet become an integral part of real-world design flows is that these algorithms were not integrated with widely-used state-of-the-art tools and methodologies. However, DSVS can be carried out without the need for dedicated algorithms, as indicated by the following discussion of a typical cell-library-based gate sizing method [25, 75].

6.2.1 Cell-Library-Based Gate Sizing Algorithms

At the logic level, library cells c_i can be represented by tuples of basic properties, namely the functionality F_i, the delay t_{Di}, the output signal transition time t_{TOi}, the dynamic power consumption P_i, the cell area A_i, and the input pin capacitances C_{Gi}:

$$c_i = \{F_i, t_{Di}, t_{TOi}, P_i, A_i, C_{Gi}\} \tag{6.1}$$

Cell-library-based gate sizing algorithms revert to the cell properties mentioned above when picking cells that implement certain functionalities while minimizing a cost function *COST* that evaluates the overall delay, the power, and the area of a circuit [25]. In Figure 6.2, the pseudo-code of a simplified gate sizing algorithm is shown. In the case of delay-constrained power optimization, the initial solution is a timing- and possibly area-optimized implementation of a logic network *NW*. Static timing analysis is used for calculating the timing slack. In each of the subsequent iterations (loops), all nodes n in the network *NW* are visited. For each node n, the complete set $C(n)$ of library cells c_i that implement the required functionality $F(n)$ is

$$C(n) = \{c_i | F_i = F(n)\} \quad . \tag{6.2}$$

The algorithm determines which cell c_{opt} in $C(n)$ must be used for replacing the cell $c(n)$ that currently implements the node under consideration, in order to maximize the cost reduction *DeltaCOST*. The substitution $c(n) = c_{opt}$ is then appended to a list of possible substitutions. Once all nodes have been visited, a subset of independent substitutions from the list off all possible substitutions is chosen, such that the total cost reduction in this iteration is maximized. Subsequently, the timing data is updated. If a cost reduction has resulted from that iteration and if there is still positive slack remaining, the algorithm continues with another iteration. Otherwise it stops.

```
start from timing-optimized initial solution
perform static timing analysis
loop {
  possible_substitutions = {}
  foreach n in NW {
     copt = c(n)
     DeltaCOSTopt = 0
     foreach c in C(n) {
        if DeltaCOST(c(n)=c) < DeltaCOSTopt {
          copt = c
          DeltaCOSTopt = DeltaCOST(c(n)=c)
        }
     }
     append "c(n) = copt" to possible_substitutions
  }
  apply max. independent subset of possible_substitutions
  update timing
  exit loop if no improvement achieved
  exit loop if no positive slack left
}
```

Figure 6.2: Typical cell-library-based gate sizing algorithm.

Actual implementations of such algorithms may differ with regard to delay and power modeling, the way of updating timing data, the treatment of local minima, or the way of determining maximum sets of independent substitutions. Nonetheless, the generic algorithm discussed above illustrates the general principles of cell-library-based gate sizing.

6.2.2 Exploiting Gate Sizing Algorithms for DSV Logic Synthesis

Reducing the supply voltage for a cell affects only its timing and power characteristics. Therefore, if two different supply voltages are allowed, each library cell c_i may be represented by two low and high voltage synthesis models c_{iLV} and c_{iHV}, respectively. The two models are functionally equivalent but exhibit different timing and power characteristics:

$$c_{iLV} = \{F_i, t_{DiLV}, t_{TOiLV}, P_{iLV}, A_i, C_{Gi}\} \tag{6.3}$$

$$c_{iHV} = \{F_i, t_{DiHV}, t_{TOiHV}, P_{iHV}, A_i, C_{Gi}\} \tag{6.4}$$

For DSVS, however, an additional constraint is required for preventing high voltage cells from being driven by low voltage cells, as described in Section 5.1.2. This can for instance

be accomplished by means of two additional pin connectivity properties describing the ideal input signal level (ISL) and the allowed fan-out signal levels (FSL):

$$c_i = \{F_i, t_{Di}, t_{TOi}, P_i, A_i, C_i, ISL_i, FSL_i\} \tag{6.5}$$

Allowed values for ISL are LV and HV for low and high voltage signals, respectively. The parameter FSL can take on one of the two values LV and DC for low voltage and don't care, respectively. Functionally equivalent low voltage, high voltage and level-converting (LC) cells can then be modeled as follows:

$$c_{iLV} = \{F_i, t_{DiLV}, t_{TOiLV}, P_{iLV}, A_i, C_{Gi}, LV, LV\} \tag{6.6}$$

$$c_{iHV} = \{F_i, t_{DiHV}, t_{TOiHV}, P_{iHV}, A_i, C_{Gi}, HV, DC\} \tag{6.7}$$

$$c_{iLC} = \{F_i, t_{DiLC}, t_{TOiLC}, P_{iLC}, A_{iLC}, C_{GiLC}, LV, DC\} \tag{6.8}$$

If, finally, the set $C(n)$ of candidates for substituting a current implementation $c(n)$ is restricted to Equation 6.2 if $ISL = LV$ for all cells driven by $c(n)$ and to

$$C(n) = \{c_i | F_i = F(n) \quad \text{and} \quad FSL_i = DC\} \tag{6.9}$$

otherwise, cell-library-based gate sizing algorithms, such as the one discussed above, can be exploited for performing DSVS and gate sizing simultaneously [75].

6.2.3 Modeling Standard Cells for Logic Synthesis

In typical standard cell libraries, the tuple of basic cell properties specified by Equation 6.1 is modeled as follows. The functionality F, i.e. the basic data-input-to-output behavior of combinational and sequential cells, is described by boolean equations. Special functionalities of sequential cells such as clear or preset are modeled separately by means of special library modeling constructs. The delay t_D and the output transition time t_{TO} are modeled as functions of the input transition time t_T and the output load capacitance C_{node} in the form of two-dimensional look-up tables. The same approach is used for modeling the setup and hold times of sequential cells. The gate input capacitances C_G are modeled as one constant value per pin, and the cell area A is specified as one constant value per cell. The dynamic power consumption is not directly included in the synthesis models of standard cells because that would require the switching activity to be considered in the characterization process. Instead, the energy E dissipated during a single transition is modeled in the libraries. More details on modeling the dynamic power consumption in SSV and DSV synthesis libraries follows in Section 6.5.3. A practical way of modeling the ISL and FSL pin connectivity properties in DSV libraries are also described there.

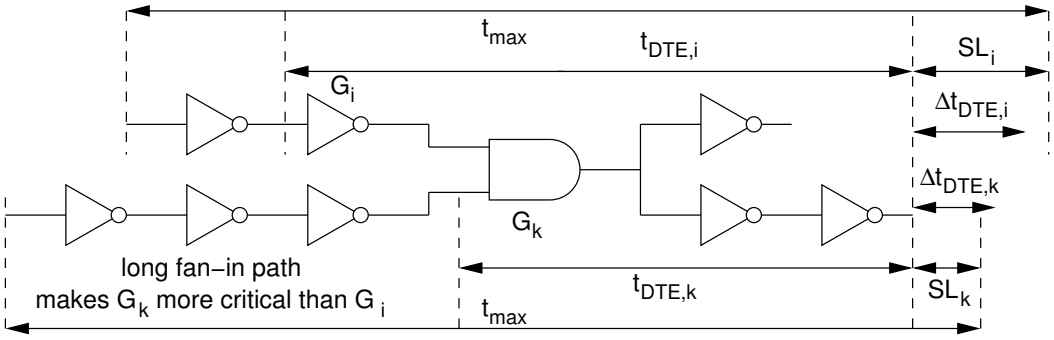

Figure 6.3: Illustration of the timing conditions for the PSEM.

6.3 Power Savings Estimation Method

The power savings that can potentially be achieved by DSVS can be predicted using Equation 5.11. The difficult part is the determination of the factors ω_i that express whether or not the supply voltage can be scaled from V_{DD} down to V_{DDL} for the i-th gate. A common approach is to interpret ω_i as the probability of the i-th gate being supplied with V_{DDL} and to use the slack of the longest path through the i-th gate as a measure of this probability, as in the type-1 and type-2 slack analysis procedures introduced in Section 7.5.3. This is, however, inaccurate and cannot yield quantitative predictions of the potential power savings. In this work, ω_i is determined by means of a more complex slack analysis procedure which leads to a simple but reasonably accurate power savings estimation method (PSEM).

Figure 6.3 illustrates the timing conditions that are evaluated in the slack analysis procedure. Suppose that all the gates in the circuit are currently supplied with V_{DD}. In order to operate the gate G_i at V_{DDL}, all gates G_k in its fan-out paths must be operated at V_{DDL} as well. This is a consequence of the level conversion issue. Scaling the supply voltage for the gate G_i and all its fan-out gates increases the delay $t_{DTE,i}$ (delay-to-endpoint, DTE) from the input of gate G_i to the endpoint of the longest path by a factor of $1+p$. With Equation 2.15, the delay increment Δt_{DTE} can be expressed as

$$\frac{\Delta t_{DTE}}{t_{DTE}} = 1 - \frac{V_{DDL}}{V_{DD}} \left(\frac{V_{DD} - V_t}{V_{DDL} - V_t} \right)^{\alpha} = p \quad . \tag{6.10}$$

A first condition that must be met is that the slack SL_i of the longest path running through the gate G_i is larger than or at least equal to the delay increment $\Delta t_{DTE,i}$. However, some gates G_k in the fan-out paths of G_i might be more critical, as illustrated in Figure 6.3. Therefore, a second condition is that the slack SL_k of all gates G_k in the fan-out paths of G_i must be larger than or equal to the respective delay-to-endpoint increment $\Delta t_{DTE,k}$. This

can be formulated as follows:

$$\omega_i = \begin{cases} 1 & : \quad \Delta t_{DTE,i} \leq SL_i \quad \text{and} \quad \Delta t_{DTE,k} \leq SL_k \quad \forall \quad G_k \text{ in fan-out of } G_i \\ 0 & : \quad \text{otherwise} \end{cases} \tag{6.11}$$

The PSEM that has been developed and used in this work is a three-step procedure. The slack analysis method described above has been implemented in TCL in such a way that it can be executed from the command line interface of a popular static timing analyzer. Just as in the CVS implementation, tool-specific TCL commands have been used for static timing analysis. The result of this first step is a list of the parameters ω_i for all N gates in the circuit. In the second step, the total dynamic power consumption P_{dyn} and the power consumption $P_{vdd,i}$ of each individual gate before voltage scaling is determined using state-of-the-art gate-level power analysis. Finally, a PERL (Practical Extraction and Report Language) program computes the potential power savings using the output of the two preceding steps and Equation 5.11 with the parameter *psc* set to zero, so as to obtain conservative estimates. For a better presentation of the results, a power savings index *PSX* shall be defined and introduced into Equation 5.11 as follows:

$$\left| \frac{\Delta P_{dyn}}{P_{dyn}} \right| = \left[1 - \left(\frac{V_{DDL}}{V_{DD}} \right)^2 \right] \cdot \sum_{i=1}^{N} \omega_i \frac{P_{vdd,i}}{P_{dyn}} =: \left[1 - \left(\frac{V_{DDL}}{V_{DD}} \right)^2 \right] \cdot PSX \tag{6.12}$$

The parameter *PSX* describes the dynamic power consumption of all scalable gates while these gates are still supplied with V_{DD}. This is a unique characteristic of each individual circuit and can be used as a measure of the optimization potential.

The main objective of developing this PSEM has been to facilitate an improved analysis of the optimization potential in the discussion of experimental results in Chapter 7. At the current stage of development, the method is restricted to the analysis of combinational circuits. An extension to sequential circuit analysis would require that the clock-to-output delays and the setup times of the sequential elements be taken into account appropriately. Since the method relies on standard tools and conventional SSV standard cell libraries, designers could then use it as a tool for predicting the effectiveness of DSVS for specific circuits before spending the effort of developing a DSV library.

6.4 Design Flow and Tools

Provided that a suitably modeled DSV library exists, delay-constrained DSV power optimization can be performed following the three-step strategy illustrated in Figure 6.4 [74, 75, 76, 78]. After reading the original design, delay-constrained logic synthesis is carried out (STEP 1). At this stage, low voltage (V_{DDL}) and level-converting cells (LC) are disabled. After capturing the switching activities $\alpha_{01,i}$ at all nodes i during logic-/gate-level

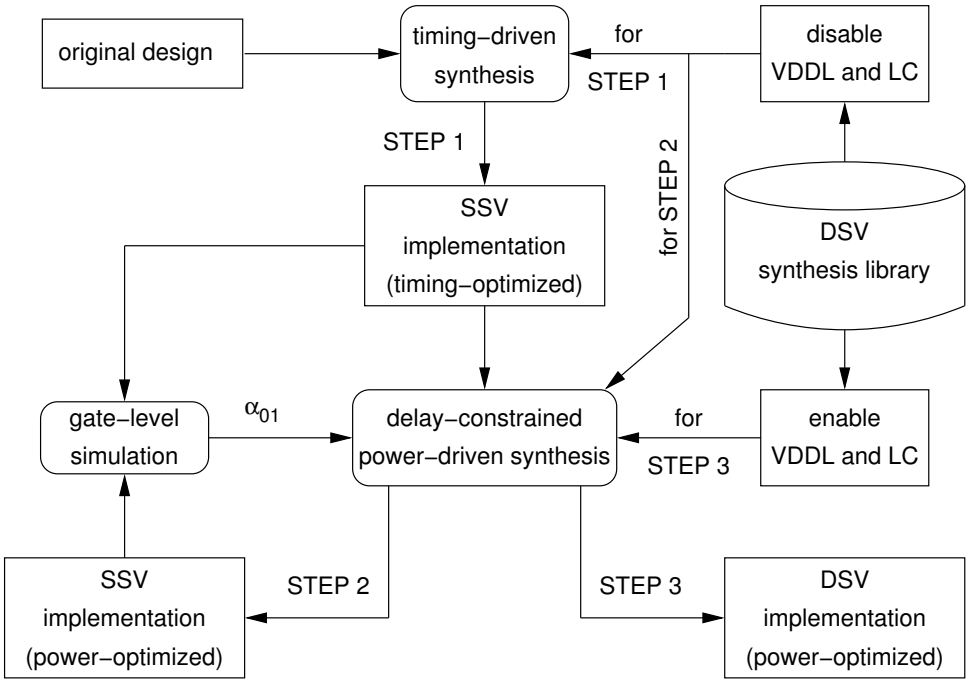

Figure 6.4: DSV logic synthesis flow.

simulation, state-of-the-art delay-constrained power optimization comprising the technology dependent techniques mentioned in Section 3.5 is carried out (STEP 2), which results in a timing- and power-optimized SSV implementation of the design. Finally, power optimization is repeated with low voltage and level-converting cells enabled (STEP 3), which leads to a timing- and power-optimized DSV implementation.

The separate SSV power optimization step (STEP 2) is actually optional. Since DSV power optimization always includes SSV optimization in this methodology, STEP 3 could be performed right after STEP 1 as well. In this study, however, STEP 2 has always been performed for comparison between the results of SSV and DSV power optimization.

The timing- and power-driven logic synthesis steps have been carried out using SYNOPSYS' POWER COMPILER. This tool is capable of minimizing power by means of a gate sizing method which behaves similarly to that discussed in Section 6.2.1. Note that the exact implementation of the gate sizing method in this particular tool is, of course, unknown. Nevertheless, the idea of exploiting the gate sizing capability for DSVS, which has been outlined in Section 6.2.2, can be realized if the DSV synthesis library is modeled in accordance with the modeling guidelines that are discussed in Section 6.5.3. A particularly important aspect is that the tool allows input and output pins of cells to be classified such that only pins of the same class are interconnected. This feature can be used for solving the level conversion issue. Since a state-of-the-art synthesis tool can be used for power optimization in this methodology, all state-of-the-art logic-level optimization techniques, including gate

up- and down-sizing and transformations of the logic structure, can be performed at the same time. Another advantage of this approach is that DSV power optimization can easily be applied to any type of circuit including complex sequential designs.

The power consumption has been analyzed at the gate-/logic-level in the pre-layout design phase using SYNOPSYS' DESIGN POWER. Pre-layout power analysis means that estimated interconnect capacitance values are used. This type of power analysis is known to be inaccurate in terms of absolute values. However, the accuracy in terms of relative values is generally considered sufficient for comparing the effect of different power optimization options. Also, it is a relatively fast method that allows a large number of different experiments to be carried out in reasonable time. For these reasons, pre-layout gate-level power analysis has been chosen in this work.

Gate-level power analysis and optimization both require the switching activities at all circuit nodes to be determined. This can be accomplished by means of RTL or gate-level simulation. The complexity of the circuits analyzed in this work is relatively low, so that full gate-level simulation can be carried out in short time. Thus, gate-level simulation using VSS (SYNOPSYS) or VERILOG-XL (CADENCE) has been preferred to the less accurate RTL-simulation-based approach.

Another important task frequently performed in this work is static timing analysis. Logic synthesis tools usually have basic static timing analysis capabilities. These are sufficient for continuously observing the validity of the results as the optimization progresses. However, advanced timing analysis strategies such as the slack distribution analysis discussed in Chapters 7 and 8 and the power savings estimation method introduced in Section 6.3 require more powerful tools. SYNOPSYS' PRIME TIME has been chosen for this work.

6.5 Dual Supply Voltage Standard Cell Libraries

The key to DSV logic synthesis exploiting gate sizing algorithms is a suitable DSV standard cell library. According to Sections 5.1.2 and 6.2.2, a DSV library must meet three requirements. Firstly, the library must contain level-converting cells. Secondly, two different low and high voltage synthesis models of each cell except for the level-converting ones must exist. Thirdly, all cells must be modeled in such a way that output pins of low voltage cells are not connected to input pins of high voltage cells.

In the remainder of this chapter, the two different DSV libraries used in this work are described. First, basic information on the fabrication processes, the supply voltage values, and the type and number of cells contained in the libraries are given. Afterwards, the design of level-converting cells is described and, finally, some important aspects regarding the modeling and the characterization of standard cells for DSV logic synthesis is discussed.

6.5.1 Technologies, Voltages, and Library Contents

A 0.25 µm CMOS DSV library. A first DSV synthesis library, which is referred to as the DSVL025 library in the remainder of this document, has been derived from a commercial standard cell library realized in STMICROELECTRONICS' 0.25 µm CMOS technology (HCMOS7). The library vendor has already provided high and low voltage synthesis library files that resulted from characterizations at supply voltages of 2.5 V and 1.8 V, respectively. The given voltage levels have been used in this work in order to avoid costly re-characterizations of the library.

The existing low and high voltage synthesis models of a number of differently sized combinational cells have been copied from the two original synthesis library files to the DSVL025 library. The selection of cells has been restricted to a subset of all cells available in the original library in order to limit the characterization effort in those parts of this work where full re-characterization has been necessary (see Section 7.5.6). The selected subset of cells does not include complex gates and gates with more than two inputs. Thus, power optimization by means of complex gate composition is not possible with this library. However, all selected cells have been made available in all the sizes that exist in the original library, so as to enable effective gate sizing. Since gate sizing trades off slack against dynamic power, just as DSVS does, these techniques directly compete. Therefore, maximization of the variety of gate sizes has been given priority over the availability of complex gates.

The library further contains simple D-flip-flop cells with non-inverting outputs Q (DFFQ). The low and high voltage models of these flip-flops have been taken from the original synthesis library files. A level-converting version (DFFQLC, see Section 6.5.2) has been included in order to enable level conversion at the outputs of combinational logic blocks.

All synthesis models copied from the original synthesis library files have been modified according to the power modeling guidelines that are discussed in Section 6.5.3. The special way of dynamic power modeling suggested in these guidelines has required to include zero delay virtual driver (ZDVD, see Section 6.5.3.3) cells in the library.

A list of all 78 cells included in the DSVL025 synthesis library is given in Table 6.1. This library has been used in conjunction with the benchmark circuits (see Chapter 7).

A 0.18 µm CMOS DSV library. A second DSV synthesis library named DSVL018 has been developed on the basis of NATIONAL SEMICONDUCTOR'S 0.18 µm CMOS technology (CMOS9). This library is named DSVL018 for reference in the remainder of this document. The original CMOSX-9 standard cell library developed by NATIONAL SEMICONDUCTOR has been characterized at supply voltages of 1.8 V and 1.3 V. The first value is the nominal supply voltage defined by the library vendor. The second value has been chosen because reducing the voltage from 1.8 V to 1.3 V has approximately the same impact on the gate delay and the power consumption as reducing the supply voltage from 2.5 V to 1.8 V in the case of the HCMOS7 technology discussed before. In other words,

Type	I/O	Drive strength (size)	Supply voltages
AND	2/1	1x, 2x, 3x, 4x	1.8 V / 2.5 V
NAND	2/1	0.5x, 1x, 2x, 3x, 4x	1.8 V / 2.5 V
OR	2/1	1x, 2x, 3x, 4x	1.8 V / 2.5 V
NOR	2/1	0.5x, 1x, 2x, 3x, 4x	1.8 V / 2.5 V
XOR	2/1	1x, 2x, 3x, 4x	1.8 V / 2.5 V
XNOR	2/1	0.5x, 1x, 2x, 3x, 4x	1.8 V / 2.5 V
INV	1/1	0.5x, 1x, 2x, 3x, 4x	1.8 V / 2.5 V
BUF	1/1	1x, 2x, 3x, 4x	1.8 V / 2.5 V
DFFQ	D, CLK / Q	1x	1.8 V / 2.5 V
DFFQLC	D, CLK / Q	1x	level converter

Table 6.1: Cells provided in the 0.25 µm DSV synthesis library (DSVL025).

the selection of supply voltage values has been made equivalent for the two DSV libraries used in this work.

All non-sequential cells contained in the original high and low voltage synthesis library files have been copied to the new DSVL018 library. This includes complex gates and gates with more than two inputs. The sequential cells included in the DSV library are scan-D-flip-flops with non-inverting outputs Q, inverting outputs QN, asynchronous clear CLR and asynchronous preset PREZ. However, the inverting outputs QN are used as dedicated scan outputs and are, hence, renamed to SO (see Section 6.5.3.4). Level-converting derivates of these flip-flops have also been designed, characterized, and included in the library (SDFFCPLC, see Section 6.5.2).

In most experiments, level conversion has been enabled only at the endpoints of combinational logic paths by means of level-converting flip-flops. In certain experiments discussed in Chapter 8, however, the effectiveness of level conversion along combinational logic paths in the proposed methodology has been investigated. For this purpose, two level-converting inverter and buffer cells (INVLC and BUFLC, see Section 6.5.2) have been designed, characterized and added to the DSVL018 library.

All synthesis models copied from the original synthesis library files have been modified according to the power modeling guidelines that are discussed in Section 6.5.3 and zero delay virtual driver cells (ZDVD, see Section 6.5.3.3) are provided.

Table 6.2 lists a subset of all 562 cells included in the DSVL018 library. This library has been used for implementing and optimizing the CR16 CompactRISC processor core (see Chapter 8).

Type	I/O	Drive strength	Supply voltages
This library includes the full set of combinational cells from the CMOSX-9 library in all the available sizes. (276 different cells in total)			1.3 V / 1.8 V
INVLC	1/1	1x	level converter
BUFLC	1/1	1x	level converter
SDFFCP	D,CLK, SD,SE, PREZ,CLR / Q,SO	2x, 4x, 8x	1.3 V / 1.8 V
SDFFCPLC	D,CLK, SD,SE, PREZ,CLR / Q,SO	2x, 4x, 8x	level converter
ZDVD	1/1	—	1.3 V / 1.8 V

Table 6.2: Cells provided in the 0.18 µm DSV synthesis library (DSVL018).

6.5.2 Level-Converting Standard Cells

Dual supply voltage standard cell libraries must contain level-converting cells in order to make transitions from V_{DDL} to V_{DD} possible. Commonly used level converters are based on the cascode voltage switch logic style [41]. While this logic style is not suitable for the design of conventional standard cells, as mentioned in Section 3.6, it has been exploited for level conversion many times. In the following paragraphs, the design of the level-converting cells included in the two DSV synthesis libraries discussed above is described.

All level-converting cells are based on the fundamental concept of the cascode voltage switch logic. Additional circuitry has been introduced in order to improve the timing characteristics and to implement special functionality such as clear and preset. Master latches from conventional flip-flops included in the original libraries have been reused in the level-converting cells. In other words, the circuit structure, the transistor dimensions, and the layout of the master latches used in conventional flip-flops and in level-converting flip-flops are identical. The complete cell layouts have been drawn in accordance with the respective process and library design rules defined by the silicon and library vendors.

Post-layout netlists have been extracted from the layouts and have been used for a full characterization of the cells. The resulting synthesis models have been included in the respective DSV synthesis library file. In the following paragraphs, the characteristics of the level-converting cells regarding the timing and the power consumption are compared with the characteristics of their low and high voltage counterparts. Regarding the timing penalty and the possible power overheads introduced by level-converting cells, the most important parameters are the gate delay t_D, the clock-to-output delay t_Q, the setup time t_{setup}, and the dynamic power consumption P_{dyn}. These parameters have been evaluated at different corners of the characterization parameter space, i.e. for different values of the input transition time t_T and the output load capacitance C_{node}, in order to find minimum, maximum

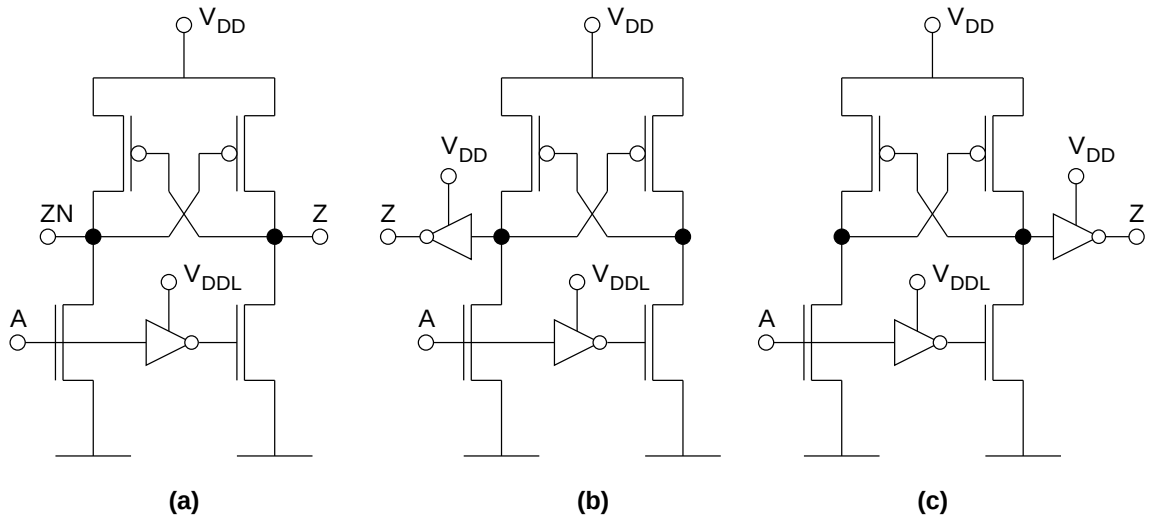

Figure 6.5: Level converters based on the cascode voltage switch logic style: (a) buffer with inverting and non-inverting output; (b) non-inverting buffer with buffered output; (c) inverter with buffered output.

and typical values. The purpose of this analysis is to provide the information required for a qualitative understanding of the behavior of level-converting cells in comparison with equivalent conventional cells.

Inverter and buffer cells. Level-converting buffers and inverters can be realized as depicted in Figure 6.5. Since the input pins A of these cells are connected only to n-channel transistors and to p-channel transistors that are source-connected to V_{DDL}, low voltage signal levels are sufficient for driving these pins. Level conversion is achieved by means of cross-coupled p-channel transistor pairs, which are source-connected to V_{DD}.

Figure 6.5a shows a simple embodiment of a level-converter with inverting (ZN) and non-inverting outputs (Z). This type of circuit is extremely difficult to design with regard to specific timing requirements under various load conditions. The reason for this is the severe impact of the output load capacitance on the feedback operation. Therefore, it is advantageous to implement extra inverters at the output nodes, so as to decouple the performance-critical internal nodes from heavy output loads, as illustrated in Figures 6.5b and c.

Two cells of this kind have been included in the DSVL018 library. The exact implementation, i.e. the circuit structure and the transistor dimensions, of the inverter cell (INVLC) and the non-inverting buffer cell (BUFLC) can be seen in Figures 6.6 and 6.7. All transistor channel lengths have been made minimal. The channel widths have been chosen such that the timing characteristics of the level-converting cells are as close as possible to those of the smallest conventional non-inverting buffer cell available in the same library when the

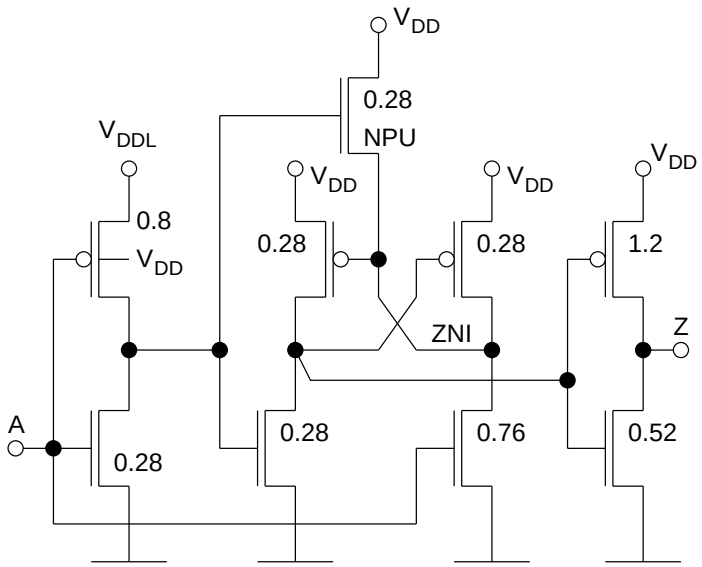

Figure 6.6: Implementation of a level-converting inverter with buffered output (INVLC) and novel pull-up technique. The channel widths are specified in units of micrometers.

	t_{DLC}/t_{DLV}				P_{LC}/P_{HV}	
with pull-up	2.0	1.6	0.9	1.0	1.2	5.3
without pull-up	3.1	1.7	0.9	1.0		
The above values have been determined at the following characterization corners:						
Input slope (t_T)	$t_{T,min}$	$t_{T,max}$	$t_{T,min}$	$t_{T,max}$	$t_{T,min}$	$t_{T,max}$
Output load (C_{node})	$C_{node,min}$		$C_{node,max}$		$C_{node,max}$	$C_{node,min}$

Table 6.3: Relative delay and dynamic power of the level-converting inverter (INVLC) compared with the delay of a low voltage (LV) buffer and the dynamic power of a high voltage (HV) buffer at different corners of the characterization parameter space.

latter is operated at the lower supply voltage V_{DDL}. Tables 6.3 and 6.4 show the delay t_{DLC} of the level-converting cells at different corners of the characterization parameter space in comparison with the delay t_{DLV} of the low voltage buffer cell. A comparison of the dynamic power consumption P_{LC} of the level-converting cells and the dynamic power P_{HV} of the high voltage buffer cell is also included in the tables.

The characterization parameter space is limited by minimum and maximum values for the input signal slope t_T and the output load capacitance C_{node}. The corner parameters ($t_{T,min/max}$ and $C_{node,min/max}$) have been chosen in accordance with the standards defined for each cell type by the vendors of the original libraries, so that the characterization results for a conventional cell and its level-converting counterpart can be directly compared.

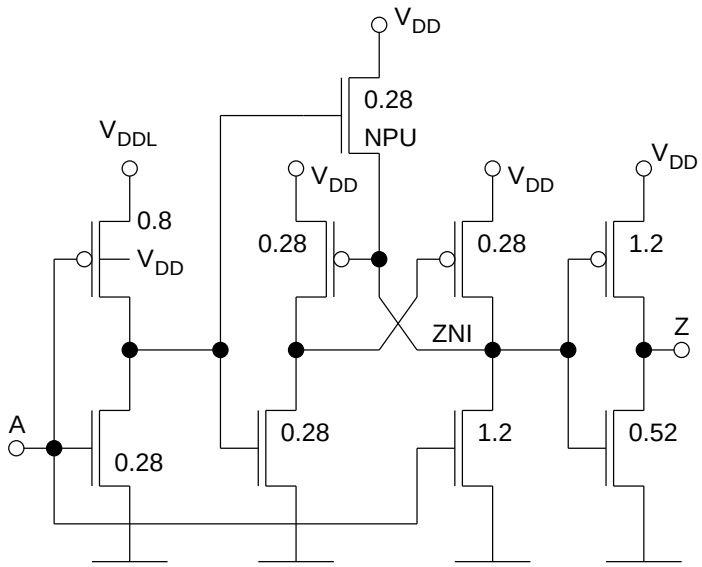

Figure 6.7: Implementation of a level-converting buffer with buffered output (BUFLC) and novel pull-up technique. The channel widths are specified in units of micrometers.

	t_{DLC}/t_{DLV}				P_{LC}/P_{HV}	
with pull-up	2.4	1.7	1.0	1.0	1.2	2.9
without pull-up	3.6	1.9	1.0	1.0		
The above values have been determined at the following characterization corners:						
Input slope (t_T)	$t_{T,min}$	$t_{T,max}$	$t_{T,min}$	$t_{T,max}$	$t_{T,min}$	$t_{T,max}$
Output load (C_{node})	$C_{node,min}$		$C_{node,max}$		$C_{node,max}$	$C_{node,min}$

Table 6.4: Relative delay and dynamic power of the level-converting buffer (BUFLC) compared with the delay of a low voltage (LV) buffer and the the dynamic power of a high voltage (HV) buffer at different corners of the characterization parameter space.

The data show that the level-converting cells have roughly the same delay as the low voltage cell for large output loads. However, it would require unfeasibly large transistors to achieve equal delay in the case of small output loads, where the delay can currently be up to 2.4 times that of the low voltage buffer. The absolute delay of the level-converting cells is in the range of 0.15 ns to 1.1 ns depending on the input slope and the output load.

Some improvement of the relatively poor timing characteristics in the case of small loads has been achieved by adding the n-channel pull-up transistors NPU to the cells. These transistors support the feedback operation of the circuits by pulling the internal node ZNI up towards the positive supply for a falling edge of the input signal. In case of a high logic state at the input pin A, the gate of NPU and the node ZNI, i.e. the source electrode of NPU,

are both pulled to ground. The pull-up transistor NPU is not conducting (off). When the logic state of the input signal changes from high to low, NPU turns on and pulls the node ZNI towards the positive supply V_{DD}. As soon as the potential at the node ZNI reaches $V_{DD} - V_t$, where V_t is the absolute value of the threshold voltage of NPU, the pull-up device turns off again. The numbers included in Tables 6.3 and 6.4 indicate that the delay of the level-converting cells and, hence, the ratio of t_{DLC} to t_{DLV} would be up to about 50% larger without this novel circuit technique being used.

The dynamic power consumption of the level-converting cells is generally higher than that of a conventional buffer cell, even if the latter is operated at the higher supply voltage. The overhead is largest in the case of long input transition times and small output loads. Under these conditions, the short-circuit power contributes significantly to the total dynamic power and the short-circuit power of the level-converting cells is significantly larger than that of the conventional buffer cell. For large loads and short input transition times, the dynamic power is clearly dominated by its capacitive component. Since the latter is the same for level-converting and conventional cells, there is only little difference in the total dynamic power under these conditions.

Layouts of these circuits have been drawn in agreement with the design rules that apply to NATIONAL SEMICONDUCTOR'S CMOS9 technology and CMOSX-9 standard cell library. Both cells have the same area which is twice that of the smallest conventional buffer cell and 2.8 times that of the smallest conventional inverter cell. The full characterization of the cell has been based on a post-layout netlist using a characterization parameter set which is typical of the original library from NATIONAL SEMICONDUCTOR.

In most experiments discussed in this study, level conversion has been restricted to the input and output nodes of combinational logic blocks by means of level-converting flip-flop cells, as explained in Section 5.1.2. The two level-converting cells described in this section have been used only in selected experiments in order to investigate the feasibility of level conversion along combinational logic paths (see Sections 8.6.2 and 8.6.3).

Flip-flop cell. Figure 6.8 shows the schematic of the most widely used static D-flip-flop structure. This type of flip-flop can be found in most standard cell libraries. Sometimes, pairs of inverters and transmission gates are replaced with tristate inverters. The circuit is composed of two latches in series: the master latch at the input and the slave latch at the output side. This flip-flop structure can easily be transformed into a level-converting one if the slave latch is replaced with a level-converter similar to those described above. These level converters behave like latches because of the positive feedback created by means of the cross-coupled p-channel transistor pairs.

Figure 6.9 shows the schematic of the level-converting D-flip-flop (DFFQLC) included in the DSVL025 library. The master latch is basically the same as that used in the corresponding conventional D-flip-flop available in the same library. The only difference is the transmission gate TG2 which has been added to provide differential input signals for the

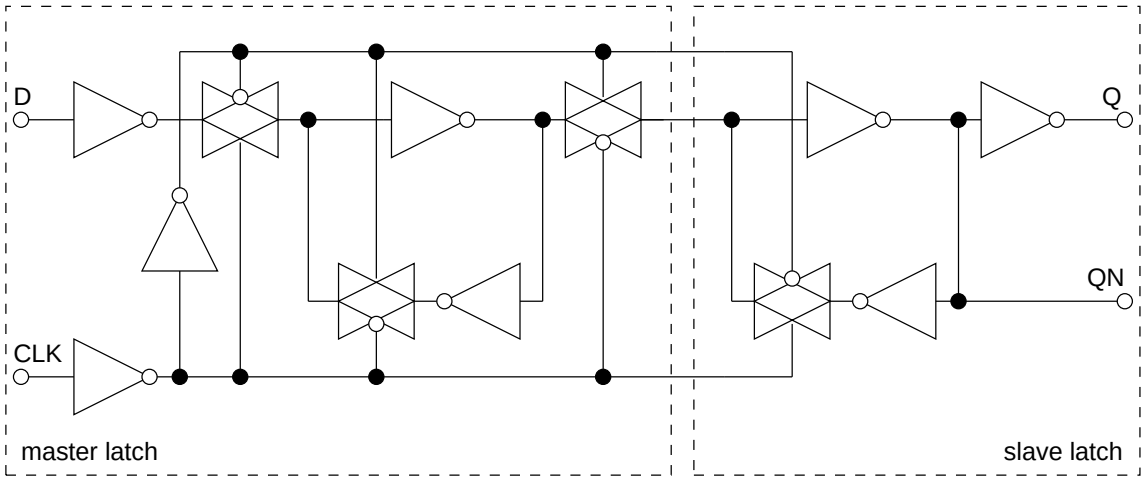

Figure 6.8: Standard D-flip-flop circuit with clock input CLK, data input D, non-inverting output Q and inverting output QN.

slave latch. The level-converting slave latch is a slightly modified buffer-type cascode voltage switch level converter. The transistors N3 and N4 have been added in order to prevent the gates of N1 and N2 from floating when the clock is zero separating the slave latch from the master latch via the transmission gates TG1 and TG2. The slave latch is supplied with V_{DD}. This flip-flop structure is the same as the one used by Usami et al. [116]. Note that if a library includes low power flip-flops similar to that introduced in Section 3.6, the same circuits can serve as level-converting cells if the slave latches are supplied with V_{DDL}.

In the slave latch of the flip-flop cell depicted in Figure 6.9, all transistor channel lengths are minimal. The channel widths have been chosen such that the timing of the flip-flop is as close as possible to the timing of the corresponding conventional flip-flop when the latter is operated at the lower supply voltage V_{DDL}. The exact values can be seen in the figure. The dimensions of the transistors used in the master latch are the same as in the conventional flip-flop designed by STMICROELECTRONICS.

Some data comparing the timing characteristics of the level-converting flip-flop with the characteristics of conventional flip-flops at different corners of the characterization parameter space is provided in Table 6.5. The clock-to-output delay t_{QLC} of the level-converting flip-flop is the same as the delay t_{QLV} of the low voltage flip-flop and 1.5 times the delay t_{QHV} of the high voltage cell. In absolute values, t_{QLC} is in the range of 0.3 ns to 1.2 ns depending on the input signal transition time and the output load capacitance. The setup time $t_{setupLC}$ is generally larger than that of the conventional cells. However, the larger the input signal transition time, the smaller is this overhead. The setup time of the level-converting flip-flop is on the order of 0.1 ns to 0.4 ns depending on the input signal transition time which is the slope of the clock signal in this case.

Regarding the dynamic power consumption, a comparison with the conventional high volt-

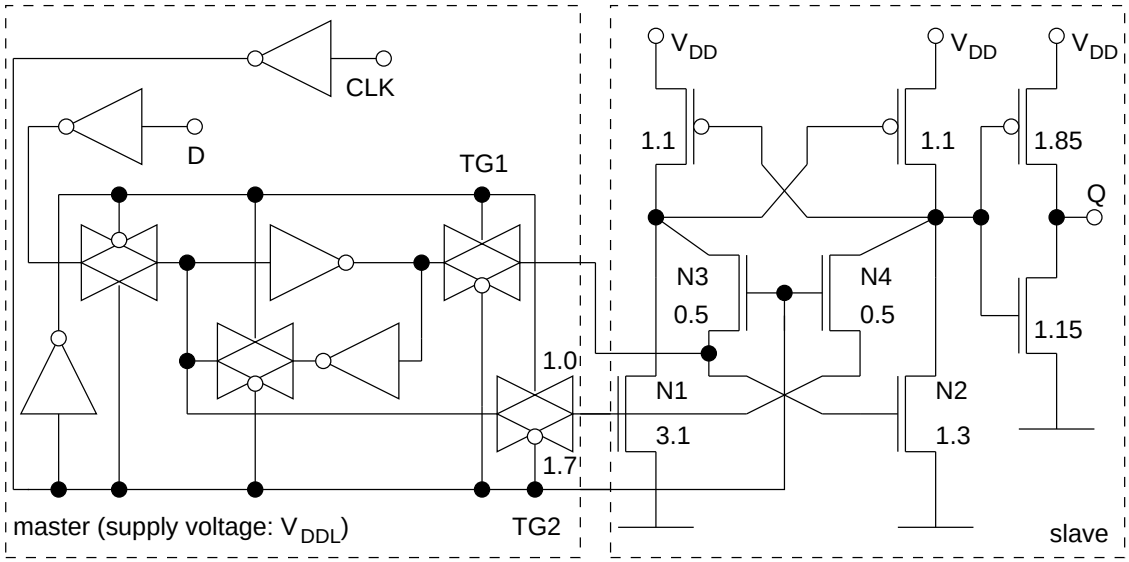

Figure 6.9: Level-converting D-flip-flop with clock input CLK, data input D and non-inverting output Q (DFFQLC). The channel widths are specified in units of micrometers.

Supply: (<XX>)	$t_{QLC}/t_{Q<XX>}$	$t_{setupLC}/t_{setup<XX>}$		$P_{LC}/P_{<XX>}$	
HV	1.5	3.3	1.6	1.3	0.9
LV	1.0	2.3	1.2	2.5	2.0
The above values have been determined at the following characterization corners:					
Input slope (t_T)	any value	$t_{T,min}$	$t_{T,max}$	$t_{T,min}$	$t_{T,max}$
Output load (C_{node})	any value	—	—	$C_{node,min}$	$C_{node,max}$

Table 6.5: Timing characteristics and dynamic power of the level-converting D-flip-flop cell (DFFQLC) compared with the conventional D-flip-flop (DFFQ) operated at low voltage (LV) or high voltage (HV).

age flip flop is of particular interest. A large ratio of P_{LC} to P_{HV} could cause a large power overhead in the clock voltage scaling scheme where all high voltage flip-flops are replaced with level-converting ones. However, according to the data included in Table 6.5, the level-converting flip-flop consumes only little more power in the case of small loads and short input transition times. For larger slopes and loads, the level-converting cell can even be more power efficient than the conventional cell. Obviously, this data does not allow to draw a clear conclusion regarding a possible power overhead caused by massive use of level-converting flip-flop cells as required for clock voltage scaling.

A layout of this circuit has been drawn in agreement with the design rules that apply to STMICROELECTRONICS'S HCMOS7 technology and standard cell library. The size of the

level-converting cell is roughly twice the size of the corresponding conventional cell. The full characterization of the cell has been based on a post-layout netlist using a characterization parameter set which is typical of the original library from STMICROELECTRONICS.

Scan-flip-flop cells. Figure 6.10 shows the circuit structure of a novel level-converting scan-D-flip-flop (SDFFCPLC) with a non-inverting output Q, an inverting dedicated scan output SO, an asynchronous preset input PREZ, and an asynchronous clear input CLR. This type of cell has been included in the DSVL018 library in three different sizes. The circuits are composed of master and slave latches that are supplied with V_{DDL} and V_{DD}, respectively. In contrast to the level-converting flip-flop described before, two n-channel pull-up transistors NPU1 and NPU2 have been added to the slave latch. This improves the delay by as much as 15% without increasing the cell area noticeably. Furthermore, the slave latch contains additional transistors that implement the preset and clear functionalities.

The master latch used in all three level-converting cells is identical to the one used in the smallest conventional flip-flop cell (SDFFCPX2). This means that both the circuit structure and the transistor dimensions are basically the same. Only two minor modifications have been made. Firstly, an additional transmission gate provides a second complementary input signal for the slave latch. Secondly, the n-well has been connected to V_{DD} instead of V_{DDL} so as to avoid the area overhead created by separate low and high voltage wells. A block diagram of the master latch is shown in the upper left corner of Figure 6.10. It is composed of a conventional D-latch with clock (CLK), clear (CLR) and preset (PREZ) inputs. The input multiplexer feeds data samples either from the data input D or from the scan data input SD to the data input of the latch. At the output side, the latch provides two complementary data signals QI and QNI, the inverted clock signal CLKN, and the inverted preset signal PREZN, which are the input signals for the slave latch.

The channel lengths of all transistors in the slave latches have been made minimal. The channel widths have been chosen such that the timing of each of the three level-converting cells (SDFFCPLCX2, SDFFCPLCX4, SDFFCPLCX8) is as close as possible to the corresponding conventional cell (SDFFCPX2, SDFFCPX4, SDFFCPX8) when the latter is operated at the higher supply voltage. The values are given in Table 6.6.

In Table 6.7, the timing and the power characteristics of the level-converting cells are compared with the characteristics of the conventional cells operated at the higher supply voltage. The level-converting cells have larger clock-to-output delays than the conventional cells in the case of small load capacitances. The largest delay overhead can be observed if the load is small and the input transition time is large. For large load capacitances, however, the level-converting cells have slightly shorter delays. Their setup times are large in comparison with the conventional cells in the case of small transition times at the clock input. For large input transitions, however, the level-converting cells have shorter setup times. In absolute values, the clock-to-output delays of the level-converting cells are in the range of 0.2 ns to 1.0 ns depending on the input transition time and the load capacitance. Their

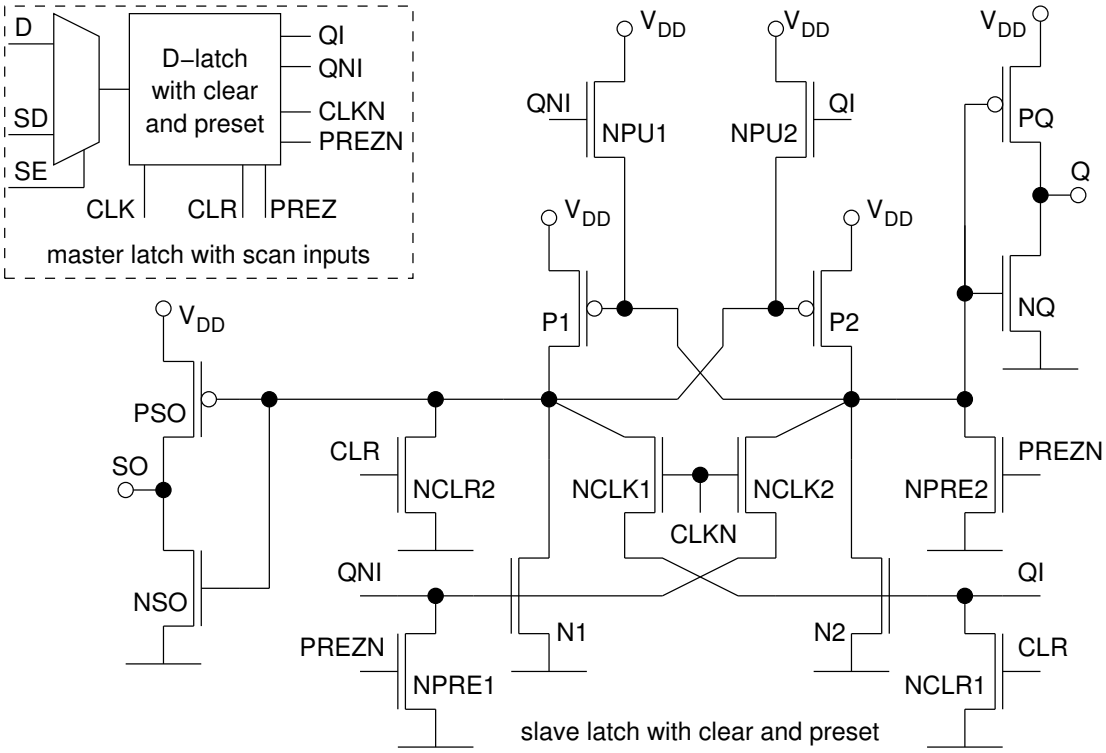

Figure 6.10: Novel level-converting scan-D-flip-flop with non-inverting output Q and dedicated inverting scan output SO (SDFFCPLC).

setup times are in the range of 0.3 ns to 0.7 ns. The level-converting flip-flops consume the same amount of dynamic power as the conventional cells in the case of small input transition times and large output loads. In all other cases, the level-converting cells are even slightly more power efficient. According to this data, replacing a high voltage flip-flop with its level-converting counterpart does not create a power overhead.

Layouts have been drawn for all three level-converting cells in accordance with the design rules that apply to NATIONAL SEMICONDUCTOR'S CMOS9 technology and CMOSX-9 standard cell library. The level-converting cells are between 17% (SDFFCPLCX8) and 28% (SDFFCPX2, SDFFCPX4) larger than the corresponding conventional cells. The master latches used in the scan-flip-flops take up a large portion of the total cell area, so that the overhead caused by the level-converting slave latches is relatively small. Therefore, the area overhead of the SDFFCPLC cells is small compared with the overhead of the DFFQLC cell included in the DSVL025 library. The full characterization of the cell has been based on a post-layout netlist using a characterization parameter set which is typical of the original library from NATIONAL SEMICONDUCTOR.

	SDFFCPLCX2	SDFFCPLCX4	SDFFCPLCX8
N1/2	2.00 µm	2.00 µm	3.00 µm
P1/2	1.40 µm	1.40 µm	2.00 µm
NPU1/2	1.20 µm	1.20 µm	2.00 µm
NCLR1	0.28 µm	0.28 µm	0.28 µm
NCLR2	2.00 µm	2.00 µm	2.00 µm
NPRE1	0.28 µm	0.28 µm	0.28 µm
NPRE2	1.20 µm	1.20 µm	2.00 µm
NCLK1/2	0.28 µm	0.28 µm	0.28 µm
NQ	2.00 µm	4.00 µm	6.00 µm
PQ	3.00 µm	6.00 µm	10.0 µm
NSO	1.40 µm	2.80 µm	4.20 µm
PSO	2.20 µm	4.28 µm	8.00 µm

Table 6.6: Channel widths for the transistors used in the slave latches of the level-converting scan-D-flip-flops with three different driving strengths.

	t_{QLC}/t_{QHV}		$t_{setupLC}/t_{setupHV}$		P_{LC}/P_{HV}	
SDFFCP(LC)X2	3.3	0.8	1.4	0.6	1.0	0.6
SDFFCP(LC)X4	3.2	0.8	1.7	0.6	1.0	0.8
SDFFCP(LC)X8	2.8	0.9	1.7	0.6	1.0	0.8
The above values have been determined at the following characterization corners:						
Input slope (t_T)	$t_{T,max}$	$t_{T,min}$	$t_{T,min}$	$t_{T,max}$	$t_{T,min}$	$t_{T,max}$
Output load (C_{node})	$C_{node,min}$	$C_{node,max}$	—	—	$C_{node,max}$	$C_{node,min}$

Table 6.7: Timing characteristics and dynamic power of the level-converting scan-D-flip-flop cells compared with the conventional scan-D-flip-flops. The latter are operated at the higher supply voltage.

6.5.3 Library Modeling and Characterization

6.5.3.1 Power Modeling, Characterization and Analysis

Dynamic power modeling. The common way of handling the dynamic power consumption for the purpose of standard cell library modeling requires that the cell-internal dynamic power be distinguished from the external dynamic power.

The external dynamic power is the capacitive component associated with the external interconnect capacitance and the sum of all gate input capacitances presented to the output pin of the cell. This is normally not included in the synthesis library. Instead, tools for power-driven logic synthesis and gate-level power analysis use estimated or extracted interconnect capacitances together with gate input capacitance values modeled in the library for calculating the external capacitive switching power from Equation 2.20.

The cell-internal power includes the short-circuit component and a capacitive component which is due to all the cell-internal device and interconnect capacitances except for the gate input capacitances. Regarding the cell-internal power associated with a signal transition occurring at the output node of a CMOS gate, rising and falling transitions are distinguished (see Figure 2.2) and modeled by means of two separate look-up tables (LUT) in the synthesis library (see the `rise_power` and `fall_power` attributes in Figure 6.12). These tables are two-dimensional and indexed with the output load capacitance C_{node} and the input signal transition time t_T [1]. As mentioned in Section 6.2.3, the synthesis models of standard cells include switching energy rather than dynamic power values. Each value contained in the cell-internal power LUTs represents the energy E_{intR} or E_{intF} dissipated during a single rising or a falling transition, respectively.

To obtain expressions for the total energy drawn from the supply, the energy needed to charge the external load capacitance C_{node} (see Section 2.3.1) must be added to the cell-internal switching energy associated with a rising edge at the output node:

$$E_{totR} = E_{intR}(t_T; C_{node}) + C_{node} V_{DD}^2 \tag{6.13}$$

$$E_{totF} = E_{intF}(t_T; C_{node}) \tag{6.14}$$

Dynamic power characterization. When standard cells are characterized for dynamic power consumption, the energy values E_{totR} and E_{totF} are measured by means of transistor-level simulation[1] for different values of the parameters C_{node} and t_T and a fixed supply voltage V_{DD}. Subsequently, the cell-internal switching energies E_{intR} and E_{intF} are obtained by subtracting the energy needed to charge the output load from the total energy E_{totR} (see Equation 6.13). These values are finally stored in the LUTs.

Dynamic power analysis. Tools for power-driven logic synthesis and gate-level power analysis determine the total energy drawn from the supply during a full switching cycle, i.e. a rising edge followed by a falling edge or vice versa, of a particular signal by taking appropriate values for the cell-internal energies associated with these transitions from the LUTs and adding $C_{node} \cdot V_{DD}^2$. The total dynamic power consumption is then calculated

[1] Examples of transistor-level simulators are SPICE (public domain), HSPICE (SYNOPSYS), ELDO (MENTOR), and SPECTRE (CADENCE).

by multiplication with the switching activity α_{01} and the clock frequency f_{clk} and, finally, summing over all N nodes in the circuit:

$$P_{dyn} = f_{clk} \sum_{i=1}^{N} \left\{ \alpha_{01,i} \left[E_{intR,i}(t_{T,i}; C_{node,i}) + E_{intF,i}(t_{T,i}; C_{node,i}) + C_{node,i} V_{DD}^2 \right] \right\} \quad (6.15)$$

This expression is equivalent to the sum of Equations 2.20 and 2.22.

6.5.3.2 Modeling DSV Libraries in the Liberty Format

Today, most synthesis libraries are available in the LIBERTY (.lib) library format. While this has originally been a proprietary format developed by SYNOPSYS, it is now a de facto industry standard that has been disclosed to the public and can be licensed free of charge[2].

LIBERTY provides everything which is needed for DSV synthesis libraries [73]. The most important library, cell and pin attributes are shown in Figures 6.11 and 6.12. Multiple power rails can be specified using a `power_supply` group in the description of the environment at the library level. These power rails can then be assigned to individual cells by means of `rail_connection` attributes. The absence of rail connections in the description of a specific cell means that the value of the `default_power_rail` attribute is to be used. If one rail connection has been specified within a cell description, the respective voltage level has to be used in conjunction with this cell. In a DSV synthesis library, for instance, the lower supply voltage V_{DDL} could be assigned to low voltage cells using `rail_connection` attributes. In contrast, high voltage cells would not need any explicit rail connection, provided that the `default_power_rail` attribute has been assigned the higher supply voltage V_{DD}.

Finally, connection classes have to be assigned to the input and output pins of all cells by means of `connection_class` attributes such that output pins of low voltage cells are not allowed to drive input pins of high voltage cells. The idea of the connection classes is that only pins that belong to the same class are allowed to be interconnected. The inputs and outputs of low voltage cells and the inputs of level-converting cells are assigned the same connection class, for instance a class named LV, while the inputs of high voltage cells are assigned a different class, HV for instance. The outputs of high voltage and level-converting cells are assigned both the LV and HV classes. This strategy is a translation of the *ISL* and *FSL* cell attributes introduced in Section 6.2.2 into the LIBERTY format.

If the logic synthesis and gate-level power analysis tools to be used support these special LIBERTY constructs, the DSV library can be modeled as described above. Support of these constructs means that the total dynamic power is actually calculated from

$$P_{dyn} = f_{clk} \sum_{i=1}^{N} \left\{ \alpha_{01,i} \left[E_{intR,i}(t_{T,i}; C_{node,i}) + E_{intF,i}(t_{T,i}; C_{node,i}) + C_{node,i} V_{DD,i}^2 \right] \right\} \quad , \quad (6.16)$$

where $V_{DD,i}$ depends on the rail connection of the cell driving the i-th node.

[2] http://www.synopsys.com/partners/tapin/lib_info.html

```
library(DSV_LIB) {
    ...
    /* multiple power rails*/
    power_supply() {
        /* VDD is associated with nom_voltage */
        default_power_rail : VDD;
        power_rail (VDDL, 1.8);
    }
    ...
    /* nominal operating conditions */
    nom_voltage : 2.5 ;
    ...
    /* synthesis models of all cells */
    cell(<name>) { ... }
    ...
} /* end of library */
```

Figure 6.11: Excerpt from the environment part of a generic standard cell DSV synthesis library in the LIBERTY format.

```
cell(INV_LV) {
    area : 18 ;
    rail_connection (PV1, VDDL);
    pin(Z) {
        direction : output ;
        connection_class : "LV";
        function : "A'";
        internal_power() {
            rise_power(pwr_lut_template_name) {<2-dim LUT>}
            fall_power(pwr_lut_template_name) {<2-dim LUT>}
        } /* end of internal power */
        timing() { ... }
    } /* end of pin(Z) */
    pin(A) {
        direction : input ;
        connection_class : "LV";
    } /* end of pin(A) */
} /* end of cell */
```

Figure 6.12: Excerpt from a generic low voltage standard cell synthesis model in the LIBERTY format.

6.5.3.3 Modeling the Total Dynamic Power Using LUTs

An alternative approach makes DSV library modeling possible even if the logic synthesis and gate-level power analysis tools do not support `power_supply`, `power_rail` and `rail_connection` attributes [73].

This approach is based on a modified characterization procedure. Instead of subtracting the energy needed to charge the output load, the total energy as given by Equations 6.13 and 6.14 is stored in the internal power LUTs. In order to prevent the capacitive switching energy from being counted twice, the rightmost term in Equation 6.15, i.e. the explicit calculation of $C_{node} \cdot V_{DD}^2$, has to be eliminated. This can usually be accomplished by setting the nominal supply voltage value to zero in the synthesis library environment.

A drawback of this approach is that the switching power associated with the primary input nets, including the clock network, is not taken into account due to the absence of driving cells. Therefore, zero-delay virtual driver cells (ZDVD) must be inserted in the fan-out of all input ports. The synthesis models of these cells provide the LUTs that are required for determining the capacitive switching power associated with the primary input nets. Their functionality is that of a non-inverting buffer, and all their other properties and characteristics are ideal. This means that ZDVDs have no delay, exhibit no short-circuit power, and do not introduce any additional capacitances. These cells can be inserted right before power optimization and analysis and must be removed thereafter.

This power modeling approach does still require connection classes to be assigned to the input and output pins of the cells. Consequently, the `connection_class` attribute or an equivalent mechanism must be supported by the synthesis tool.

The tools used in this work (see Section 6.4) do not support multiple power rails. Thus, the LUT-based power modeling approach introduced in this section had to be used.

6.5.3.4 Modeling Scan-Flip-Flop Cells

Design for scan testability in DSV logic synthesis usually requires level-converters to be used in the scan chains as illustrated in Figure 8.7. In this work, this has been accomplished in a two-step procedure. The initial scan chain synthesis has been carried out right after the timing-driven logic synthesis without regarding the level conversion issue. Subsequently, level converters have been inserted where necessary. These level converters introduce additional delay into the scan chains. Therefore, the scan chains must be separated from all functionally relevant logic paths. Under this condition, level converter insertion in the scan chains affects only the circuit area while the overall timing remains unchanged. The reason is that scan chains are so-called false paths, i.e. they are not subject to timing constraints.

The scan chain separation can be accomplished by modeling the scan-flip-flops in the synthesis library in such a way that the scan data input pin SD and one of the output pins are

used exclusively for scan chain synthesis. In this work, the inverting outputs QN of the scan-flip-flops are used as dedicated scan output pins and are, therefore, renamed to SO, as mentioned before. In the LIBERTY library format, scan-flip-flops with dedicated scan input and output pins can be modeled as shown in Figures 6.13 and 6.14. The synthesis model of a scan-flip-flop is composed of two parts. The first part describes the general behavior of the cell just as for any other type of flip-flop (see Figure 6.13). The second part is enclosed in the `test_cell` group and provides additional information which is relevant only for the scan chain synthesis (see Figure 6.14). As can be seen from the figures, the special purposes of the scan enable pin SE, the scan data input pin SD, and the scan output pin SO are specified using the `signal_type` attribute in the `test_cell` group. The `test_output_only` attribute characterizes the output pin SO as a dedicated scan output pin.

The above modeling guidelines are usually sufficient to achieve scan chain separation in the initial scan chain synthesis. However, depending on the capabilities of the logic synthesis tool, the scan output pins may be used as functional outputs again in subsequent power-driven re-synthesis steps, which has been the case in this work. Therefore, the scan data input pins SD and the scan output pins SO have been assigned a special connection class as shown in Figure 6.13, so as to guarantee that only these pins are interconnected.

6.5.3.5 Characterization of DSV Libraries

Characterization tools. The public domain software package GSPICE[3] has been used for characterizing standard cells in this work. GSPICE is a SPICE pre- and post-processor and works with any SPICE-like circuit simulator that generates waveform data files compatible with either HSPICE (SYNOPSYS) or ELDO (MENTOR). The output of a GSPICE run is the synthesis model of a standard cell in the LIBERTY format.

When the software is used as is, only the timing characteristics and the gate input capacitances of conventional gates and flip-flops can be obtained [33]. In order to be useful in this work, the capabilities of the tool have been extended to dynamic power characterization and to the characterization of level-converting cells.

Input signal levels. Special attention has been paid to the input signal levels applied to the low voltage input pins of low voltage and level-converting cells during the characterization process. In DSV circuits, as explained in Section 5.1.2, low voltage input pins may be driven by low voltage or high voltage output pins. For the characterization of low voltage and level-converting cells, it is important to decide whether to use low or high input signal levels.

[3] http://www.veripool.com/gspice.html

```
cell(SDFFCPLCX2) {
    test_cell() { ... }
    pin(SE) {
        direction : input ;
        connection_class : "HV LV";
    ... } /* end of pin SE */
    pin(SD) {
        direction : input ;
        connection_class : "SC";
    ... } /* end of pin SD */
    pin(SO) {
        direction : output ;
        function : "INQ"  ;
        test_output_only : true ;
        connection_class : "SC";
    ... } /* end of pin SO */
... } /* end of cell */
```

Figure 6.13: Excerpt from a generic scan-flip-flop synthesis model defining dedicated scan input and output pins in the LIBERTY format.

```
test_cell() {
    pin(SD) {
        direction : input;
        signal_type : "test_scan_in";
    } /* end of pin SD */
    pin(SE) {
        direction : input;
        signal_type : "test_scan_enable";
    } /* end of pin SE */
    pin(SO) {
        direction : output;
        signal_type : "test_scan_out_inverted";
        test_output_only : true ;
    } /* end of pin SO */
... } /* end of test_cell */
```

Figure 6.14: Excerpt from a test cell group defining dedicated scan input and output pins in the LIBERTY format.

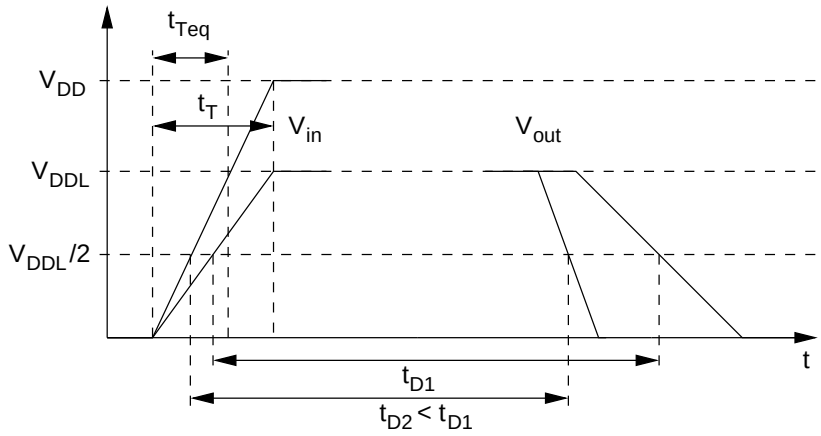

Figure 6.15: Delay characterization of low voltage cells using different input signal levels.

The waveforms in Figure 6.15 illustrate the two possible scenarios of low voltage cell characterization. For low voltage cells, the delay t_D is defined as the time it takes for the output voltage to reach $V_{DDL}/2$ after the input voltage passed the same level. If a high voltage input signal is used instead of a low voltage signal with the same transition time t_T, this is equivalent to reducing the transition time of the low voltage input signal to t_{Teq}, as depicted in the figure. Since shorter input transition times result in shorter gate delays (see Equation 2.14), using low voltage input signals for characterizing the timing of low voltage cells that are actually driven by high voltage signals yields pessimistic results. This statement is true also for the characterization of the setup and hold times of flip-flop cells. Even regarding the power consumption, low voltage input signals yield pessimistic results because an effectively shorter input signal slope results in less short-circuit power consumption (see Equation 2.22). In order to guarantee valid results under any circumstances, the low voltage and level-converting cells used in this work have been characterized with low voltage signals applied to all input pins.

Impact of layout concepts. Another aspect that must be considered in the library characterization process is the aggravated body effect that arises from the high n-well potential required by the dual power rail layout scheme (see Section 5.4). In most experiments discussed in the remainder of this document, it has been assumed that the final layout of the circuits will be based on some kind of voltage separation as in the row-by-row and split-row scenarios (see Section 5.4). Thus, the DSVL025 and DSVL018 libraries have been characterized with the n-well regions of low voltage cells connected to V_{DDL}. However, in order to investigate the impact of an aggravated body effect on the DSV logic synthesis results (see Section 7.5.6), the DSVL025 library has been re-characterized with the n-well regions of low voltage cells connected to V_{DD}.

Chapter 7

Characteristics of DSV Logic Synthesis

The DSV logic synthesis methodology introduced in the previous chapter has been used for investigating the potential and the limitations of DSVS. These investigations were motivated by the shortcomings of related work (see Section 5.3), where DSVS was carried out under unrealistic conditions.

In the experiments discussed in this chapter, DSV power optimization has been applied to various combinational and sequential circuits in a conventional design environment using standard tools for timing-driven logic synthesis and logic-level power optimization. This approach enables a more realistic judgment on the optimization potential of the DSVS technique.

7.1 Fundamental Parameters

With DSVS, positive timing slack can be traded off against dynamic power consumption and, hence, a power reduction can be achieved only if timing slack exists in the circuit to be optimized. In fact, some researchers generally applied DSVS to circuits that were not subject to the strictest timing constraints in order to assure the existence of a sufficient amount of slack [113, 115, 116, 122]. Other researchers carried out DSVS under various timing constraints. They observed an increase in the amount of power reduction due to DSVS as they relaxed the constraints [20].

These investigations were conducted in non-standard synthesis environments. Mostly, experimental tools such as SIS were used for timing-driven logic synthesis – only Usami et al. used a standard tool for this task – and state-of-the-art logic-level power optimization was not considered to a realistic extent in any case. However, the characteristics of the timing-driven synthesis, the strictness of the timing constraints, and the use of additional power optimization techniques have a large impact on the amount of available slack and, thus, on the optimization potential. This motivates further investigation of the potential

and the limitations of DSVS under varying timing constraints within a real-world synthesis environment.

The second important characteristic of DSVS is the impact of the choice of supply voltages on the power reduction [20, 113, 115, 124]. If the difference between V_{DD} and V_{DDL} is small, then a larger number of cells may be operated at V_{DDL} in order to fully exploit the available slack, but the power reduction per cell is small. If V_{DDL} is much lower than V_{DD}, the power reduction that can be achieved per cell is larger but fewer cells may be operated at V_{DDL}. In other words, the total power reduction may be small, either because of a small power reduction per cell or because of a small number of cells being operated at V_{DDL}, if V_{DDL} is made too large or too small, respectively.

Usually, an optimal V_{DDL} exists for a given value of V_{DD}. This optimum, however, depends largely on the circuit to be optimized [20]. Today, the only known way of optimizing V_{DDL} is to carry out DSVS for various values of V_{DDL} and choose the value that leads to the largest power reduction. This can be a very time consuming and costly procedure, particularly in a state-of-the-art logic synthesis environment, where a full characterization of the standard-cell-library at all the possible values of V_{DDL} is required.

The methodology and the tools used may have an impact on the exact value of the optimal V_{DDL} for a specific circuit subject to specific constraints. However, the facts that the power reduction due to DSVS depends on the choice of supply voltages, and that usually an optimal V_{DDL} can be found for a given V_{DD}, can be considered independent of methodologies and tools. For this reason, an expensive re-investigation of this characteristic does not appear to be reasonable and is, therefore, not part of this study. Instead, a single pair of supply voltage values has been chosen for all experiments and a simple and inexpensive estimation method is used for judging the quality of this choice.

7.2 Benchmark Circuits

Various combinational and sequential benchmark circuits have served as test cases in the experiments described in this chapter. These circuits have been taken from the so-called MCNC benchmark set[1]. MCNC today is the legal name of the former Microelectronics Center of North Carolina. Using circuits from this benchmark set for the evaluation of logic synthesis techniques makes the results more comparable to related work.

The circuits are distributed in the form of EDIF (Electronic Design Interchange Format) gate-level netlists. The original netlists contain cells taken from a generic library that comes

[1]The MCNC benchmark set originally contained ten combinational benchmark circuits. These circuits were used in conjunction with the 1985 International Symposium on Circuits and Systems, and were therefore called ISCAS'85 benchmark circuits. In the following years, the benchmark set has been updated and extended several times. The latest release was used in conjunction with the 1993 MCNC International Workshop on Logic Synthesis (IWLS'93). The benchmark set is available from the Collaborative Benchmarking Laboratory (CBL) at North Carolina State University [23].

with the benchmark set. In this work, a subset of circuits has been translated from the generic library to the DSVL025 synthesis library.

Some information on the complexity, the timing and the functionality of these circuits is given in Tables 7.1 and 7.2. In these tables, the circuit complexities are stated in terms of upper limits for the number of gates. The delay values denote the shortest possible critical path delays. These complexity and performance limits are reached if the circuits are synthesized subject to the strictest timing constraints. The descriptions of the functionality of the circuits have been extracted from the documentation of the benchmark set.

The circuits that formed the original ISCAS'85 benchmark set (marked with an asterisk in Table 7.1) appear to reflect the characteristics, i.e. the optimization potential, of the entire collection of circuits with reasonable accuracy. Therefore, these circuits have been chosen as a representative subset in some of the investigations discussed hereafter.

Input vectors and expected output vectors for each circuit are distributed with the benchmark set. This makes gate-level simulation for verification, power analysis, and power optimization possible, despite a lack of detailed information on each circuits' functionality. Furthermore, applying the provided input pattern renders the results more comparable to the results of related work.

7.3 Technology, Library, and Operating Conditions

The benchmark circuits have been mapped to the DSVL025 synthesis library introduced in Section 6.5.1. The library is based on STMICROELECTRONICS' 0.25 µm CMOS technology (HCMOS7) with a threshold voltage V_t of 0.5 V. The two supply voltages V_{DD} and V_{DDL} have been set to 2.5 V, which is the nominal supply voltage for the HCMOS7 technology, and 1.8 V, respectively. In order to limit the library characterization effort, all other operating conditions have been set to nominal values in this work.

7.4 Optimization Strategies and Constraints

Single and dual supply voltage power optimization. Three different strategies that have been used for optimizing the benchmark circuits are depicted in Figure 7.1. Each strategy is divided into two phases: the timing-driven synthesis (STEP1x) and the power optimization (STEP 2x). In the first and the second strategy (see left and middle columns in the figure), the original designs (START) are first optimized under delay and area constraints (STEP 1A). Power optimization is not enabled and only one supply voltage is used. The result of this first phase is a set of timing- and area-optimized SSV gate-level netlists. When clock voltage scaling is to be used in the DSV power optimization step, it must already be enabled in the timing-driven synthesis because of its potential impact on the performance.

	Inputs	Outputs	No. of gates	Delay in ns	Function
alu2	10	6	≤ 444	≥ 1.86	ALU
alu4	14	8	≤ 1238	≥ 1.29	ALU
apex6	135	99	≤ 912	≥ 0.90	logic
apex7	49	37	≤ 375	≥ 0.85	logic
b9	41	21	≤ 189	≥ 0.41	logic
c432(*)	36	7	≤ 243	≥ 1.95	priority decoder
c499(*)	41	32	≤ 310	≥ 1.74	error correcting
c880(*)	60	26	≤ 524	≥ 1.54	ALU and control
c1355(*)	41	32	≤ 270	≥ 1.86	error correcting
c1908(*)	33	25	≤ 383	≥ 2.67	error correcting
c2670(*)	233	140	≤ 718	≥ 1.57	ALU and control
c3540(*)	50	22	≤ 1316	≥ 3.30	ALU and control
c5315(*)	178	123	≤ 1703	≥ 2.48	ALU and selector
c6288(*)	32	32	≤ 3790	≥ 8.96	16-bit multiplier
c7552(*)	207	108	≤ 1917	≥ 2.42	ALU and control
dalu	75	16	≤ 880	≥ 1.60	ALU
des	256	245	≤ 4167	≥ 1.79	data encryption
i10	257	224	≤ 2449	≥ 3.27	logic
i5	133	66	≤ 497	≥ 0.71	logic
lal	26	19	≤ 150	≥ 0.48	logic
my_adder	33	17	≤ 378	≥ 1.17	adder
pair	173	137	≤ 2091	≥ 1.43	logic
rot	135	107	≤ 911	≥ 1.28	logic
term1	34	10	≤ 306	≥ 0.78	logic
vda	17	39	≤ 898	≥ 0.93	logic
x1	51	35	≤ 408	≥ 0.52	logic
x3	135	99	≤ 1004	≥ 0.90	logic
x4	94	71	≤ 541	≥ 0.78	logic

Table 7.1: Selection of combinational MCNC benchmark circuits. The circuits contained in the early ISCAS'85 benchmark set are marked with an asterisk(*).

	Inputs	Outputs	No. of cells	No. of FF	Delay in ns	Function
bigkey	262	197	≤ 3968	224	≥ 1.51	key encryption
clma	382	82	≤ 10104	33	≥ 4.33	bus interface
mm4a	7	4	≤ 228	12	≥ 1.82	minmax
mm30a	33	30	≤ 2086	90	≥ 7.81	minmax
mult16a	17	1	≤ 353	16	≥ 2.40	multiplier
s344	9	11	≤ 228	15	≥ 1.24	4-bit multiplier
s349	9	11	≤ 198	15	≥ 1.30	4-bit multiplier
s382	3	6	≤ 203	21	≥ 1.15	controller
s444	3	6	≤ 201	21	≥ 1.23	controller
s526	3	6	≤ 249	21	≥ 1.18	controller
s641	35	24	≤ 278	19	≥ 1.42	PLD
s713	35	23	≤ 316	19	≥ 1.42	PLD
s820	18	19	≤ 362	5	≥ 1.44	PLD
s832	18	19	≤ 410	5	≥ 1.50	PLD
s1196	14	14	≤ 658	18	≥ 1.52	logic
s1488	8	19	≤ 757	6	≥ 1.48	controller
s1494	8	19	≤ 793	6	≥ 1.51	controller
s9234.1	36	39	≤ 1151	135	≥ 2.11	logic
s38417	28	106	≤ 12906	1465	≥ 3.71	logic
sbc	40	56	≤ 849	27	≥ 1.29	bus controller

Table 7.2: Selection of sequential circuits from the MCNC benchmark set.

This means that level-converting flip-flops have to be used instead of their high voltage counterparts. In the second phase, SSV or DSV power optimization (STEP 2A or STEP 2B) subject to the same delay constraints as in the initial timing-driven synthesis are carried out. Note that, in this work, DSV power optimization (STEP 2B) means simultaneous use of DSVS and SSV techniques.

Global supply voltage scaling (GSVS). When the highest performance is not required, GSVS can be used as an alternative to DSVS under relaxed timing constraints (see Section 4.2). In Section 7.5.5, these two optimization strategies are compared. This comparison is based on a number of experiments where GSVS has been carried out as depicted in the rightmost column in Figure 7.1. In this third strategy, the timing-driven synthesis (STEP 1B) is subject to strict but not necessarily the strictest delay constraints. The

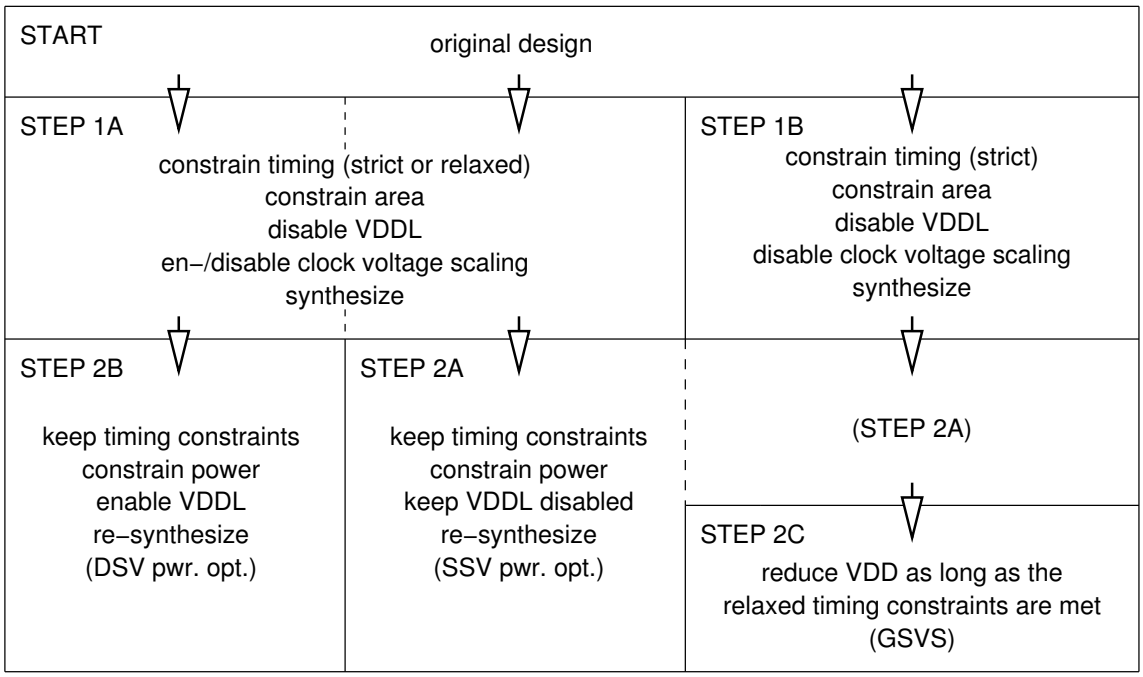

Figure 7.1: Timing and power optimization strategies.

dynamic power consumption of the resulting timing- and area-optimized SSV implementations is then minimized using power-driven SSV logic synthesis (STEP 2A). The same strict delay constraints are specified in these two synthesis steps. Finally, the global supply voltage is lowered as far as possible without violating the relaxed timing constraints (STEP 2C). Since the final result of GSVS is an SSV implementation, clock voltage scaling is generally disabled in this strategy.

Constraints. High performance is a common objective in IC design. Therefore, power optimization subject to the strictest or moderately relaxed timing constraints is a realistic task. However, even if strict timing constraints are imposed on a complex sequential design, usually the majority of combinational blocks therein are non-critical. Thus, relaxed timing constraints can be considered typical of purely combinational sub-circuits.

In the experiments discussed in this chapter, the shortest possible critical path delays of all circuits have been determined by timing-driven synthesis using zero-delay constraints. In the case of sequential circuits, these values have been used as constraints in the power-driven logic synthesis. In another series of experiments, the constraints have been relaxed to 1.2 times the shortest possible delays. This means that the sequential circuits have been optimized under the strictest and under moderately relaxed timing constraints. In the case of purely combinational circuits, the critical path delays have been constrained to 1.2 times the shortest possible delays in most cases. In some experiments, additional more or less

relaxed constraints have been specified, i.e. the delays have been constrained to 1.1, 1.2, 1.35, and 1.5 times the shortest possible delays. Thus, the combinational circuits have generally been subject to relaxed timing constraints.

The area and dynamic power constraints have been set to zero. Since the cost function gives priority to timing over power and to power over area, the power consumption has been optimized without degrading the performance and the area has been minimized without increasing the delay or the power consumption.

7.5 Optimization of Combinational Circuits

7.5.1 Single and Dual Supply Voltage Power Optimization

The power consumption of all 28 combinational benchmark circuits listed in Table 7.1 has been optimized, firstly, using state-of-the-art power-driven logic synthesis (SSV power optimization) and, secondly, using the DSV logic synthesis methodology, i.e. DSVS combined with SSV optimization. The timing constraints have been moderately relaxed to 1.2 times the shortest possible critical path delays. The results are summarized in Table 7.3.

The second column shows the power reduction achieved through SSV power optimization. Compared with the results of timing-driven synthesis, the dynamic power consumption has been reduced by 18% on average and by up to 33% in the best case (c1355). Note that, although the initial timing-driven synthesis has not included explicit power optimization, the power has indirectly been optimized along with the area. This can be seen from the results of experiments where area constraints have been omitted from the timing-driven synthesis (see Table B.1 in the appendix).

The third column of Table 7.3 shows that the average power reduction has increased from 18% to 23% due to DSVS being used in addition to SSV power optimization. In the best case (x3), DSVS has increased the power reduction from 23% to 38%.

In the fourth column, the dynamic power consumption after DSV optimization is compared with the power consumption after SSV power optimization. In other words, the numbers indicate the additional benefit of DSVS. On average, the power consumption is 7% lower when DSVS has been used. In the best case (x3), the improvement has been 20%.

7.5.2 Comparison with Related Work

Obstacles to a comparison with previously published results. As mentioned in Section 5.3, Usami et al. used the CVS method on submodules of various real applications designed in different technology generations. In any case, the choice of V_{DDL} was carefully optimized. A state-of-the-art synthesis tool was used for the timing-driven synthesis.

P_{dyn} after ...	SSV pwr. opt.	DSV$^{(*)}$ pwr. opt.	DSVS	CVS
comp. with ...	before pwr. opt.		after SSV pwr. opt.	
alu2	-15%	-16%	-1%	±0%
alu4	-15%	-16%	-3%	-1%
apex6	-21%	-29%	-10%	-8%
apex7	-19%	-26%	-8%	-9%
b9	-19%	-23%	-5%	-3%
c432	-18%	-21%	-3%	±0%
c499	-18%	-20%	-2%	±0%
c880	-12%	-22%	-12%	-4%
c1355	-33%	-33%	±0%	±0%
c1908	-17%	-23%	-7%	-6%
c2670	-21%	-23%	-3%	-2%
c3540	-17%	-21%	-5%	-1%
c5315	-18%	-28%	-12%	-9%
c6288	-8%	-14%	-6%	-2%
c7552	-8%	-16%	-9%	-5%
dalu	-20%	-22%	-3%	±0%
des	-12%	-15%	-6%	-3%
i10	-21%	-32%	-14%	-11%
i5	-19%	-26%	-5%	-6%
lal	-18%	-20%	-2%	-2%
my_adder	-15%	-22%	-13%	-7%
pair	-21%	-29%	-9%	-8%
rot	-23%	-35%	-13%	-12%
term1	-14%	-18%	-5%	-1%
vda	-11%	-12%	-1%	±0%
x1	-19%	-20%	-1%	-2%
x3	-23%	-38%	-20%	-11%
x4	-23%	-30%	-12%	-8%
avg.	-18%	-23%	-7%	-4%

Table 7.3: Optimization of combinational benchmarks. Critical path delays set to 1.2 times the minimum. $^{(*)}$DSV power opt. includes both DSVS and SSV optimization.

However, according to the slack distributions, the timing constraints were excessively relaxed in most cases and state-of-the-art power-driven logic synthesis was not part of the methodology. Thus, the circumstances were very different from this work, which makes a judgment on the published results and a direct quantitative comparison with the results obtained using the methodology proposed in this work impossible.

Yeh et al. applied the Gscale algorithm to combinational MCNC benchmark circuits that were subject to moderately relaxed timing constraints. In this respect, the characteristics of DSVS were investigated under similar conditions to this study. The technology and the choice of supply voltages, however, were different. The timing-driven logic synthesis was carried out using the experimental SIS package, which tends to produce netlists that exhibit large amounts of slack even if the strictest timing constraints are specified (see Section 7.5.3), and the SSV reference designs were not optimized for power using state-of-the-art logic-level techniques. Therefore, a direct comparison of the published results with the results obtained using the methodology presented in this study is not possible.

In the most recent work by Chen et al., the DVPO algorithm was used for optimizing combinational MCNC benchmark circuits under strict and relaxed timing constraints. The technology generation and the voltages were different from this work. Particularly, the choice of V_{DDL} was carefully optimized for each individual circuit. An important observation is that, after timing-driven logic synthesis using the SIS package, the circuits exhibited significantly more slack than after timing-driven synthesis using state-of-the-art tools (see Section 7.5.3). Finally, the SSV designs that served as references were not optimized using state-of-the-art power-driven logic synthesis. For these reasons, the results published by Yet et al. are also not suitable for a direct comparison.

Comparison of results using CVS as a reference. A fair comparison with related work requires that all relevant aspects, such as the selection of circuits, the timing constraints, the technology and the library, the supply voltages, and the use of state-of-the-art power optimization techniques, be taken into account. For this reason, the CVS algorithm explained in Section 6.1 has been implemented in such a way that it fits into the synthesis environment used in this work. The CVS algorithm has been chosen for two reasons. The first reason is its simplicity, which makes an integration into the existing synthesis environment possible. The second reason is that other researchers, namely Yeh et al. and Chen et al. also used CVS as a reference.

The CVS algorithm has been applied to the SSV power optimized combinational benchmark circuits. Column five of Table 7.3 shows that the additional power reduction due to CVS has been only 4% on average and 11% at most, which is significantly less than what has been achieved using the novel DSV logic synthesis methodology proposed in this work.

Yeh et al. applied their Gscale algorithm to a set of circuits that includes those listed in Table 7.3 except for c1908 and c6288. For these 26 circuits, Gscale yielded an average power reduction of 20%. On the other hand, even the CVS algorithm yielded an average

power reduction of 12% in the experiments carried out by Yeh et al. as opposed to 4% in this study. Chen et al. obtained very similar results when they applied the DVPO and CVS methods to a different set of benchmark circuits. This large difference, i.e. 4% as opposed to 12%, is the result of different basic conditions under which DSVS has been used.

As mentioned before, Yeh et al. used DSVS in the form of the Gscale and CVS algorithms directly after timing-driven synthesis without preceding SSV power optimization. In order to make the results of this study more comparable to those published by Yeh et al., the CVS algorithm has been applied to the said subset of 26 benchmark circuits again. This time, the SSV power optimization has not been carried out before. Under these circumstances, an average power reduction of 7% – the value increases to 8% when area constraints are omitted from timing-driven synthesis – has been achieved (see Table B.2 in the appendix). This is due to the optimization potential being larger directly after timing-driven synthesis. The additional 4% to 5% power reduction reported by Yeh at el. can be attributed to the use of SIS for timing-driven synthesis. The SIS package appears to generate circuits that exhibit larger optimization potential than the output of state-of-the-art tools (see Section 7.5.3).

The above arguments do not provide a direct comparison between the results included in this chapter and those published by Yeh at al. and Chen et al. Nevertheless, the comparison of CVS used in the different synthesis environments reveals the important differences in the conditions under which DSVS has been performed. This clearly puts the published values of about 20% power reduction achieved through Gscale and DVPO into perspective.

7.5.3 Analysis of the Optimization Potential

7.5.3.1 Effectiveness of DSVS and Gate Sizing

The two most important power optimization techniques used in the DSV logic synthesis methodology are DSVS and gate sizing. Both techniques trade off slack against power consumption. In contrast to gate sizing, the applicability of DSVS is constricted by the level conversion issue, as explained in Section 5.1. Therefore, gate sizing can be expected to be more effective than DSVS. This is confirmed by the data given in Table 7.4.

Column two indicates that, on average, 64% of all cells in the DSV power optimized benchmark circuits are of minimum size. The numbers in columns three and four show that voltage scaling has been applied to 21% of all cells, and almost all low voltage cells have minimum size as well. This means that gate sizing has been preferred, while DSVS has been used primarily when gate sizing has left slack unutilized.

As explained in Section 3.5, gate sizing minimizes the capacitive power consumption P_{cap} by reducing the gate input capacitances C_G and, hence, the node capacitance C_{node} (see Equation 2.21). The interconnect capacitance C_{int} is not affected. Consequently, significant power savings through gate sizing can be expected only if the gate input capacitances

	Amount of cells with			avg. C_{int}/C_G
	min. size	low voltage	both	
alu2	57%	$< 1\%$	$< 1\%$	0.6
alu4	61%	1%	1%	0.8
apex6	67%	41%	37%	0.8
apex7	68%	29%	29%	0.5
b9	49%	29%	26%	0.4
c432	37%	8%	8%	0.3
c499	59%	0	0	0.4
c880	59%	29%	28%	0.5
c1355	81%	0	0	0.4
c1908	72%	25%	25%	0.5
c2670	72%	9%	9%	0.7
c3540	69%	7%	7%	0.8
c5315	76%	39%	38%	1.0
c6288	47%	3%	3%	1.0
c7552	72%	12%	12%	0.9
dalu	71%	6%	6%	0.8
des	53%	24%	21%	1.1
i10	84%	52%	52%	1.1
i5	73%	40%	35%	0.6
lal	56%	17%	17%	0.4
my_adder	73%	23%	21%	0.4
pair	70%	32%	31%	0.9
rot	82%	48%	56%	0.7
term1	52%	6%	5%	0.4
vda	31%	1%	1%	0.6
x1	53%	8%	8%	0.5
x3	71%	66%	58%	0.8
x4	79%	27%	27%	0.6
avg.	64%	21%	20%	0.7

Table 7.4: Properties of combinational benchmarks after DSV power optimization. Critical path delays set to 1.2 times the minimum.

account for a significant portion of the total node capacitance. The rightmost column of Table 7.4 confirms that this is true for the benchmark circuits realized in the 0.25 µm CMOS technology. On average, the ratio of C_{int} to C_G is 0.7. In other words, the gate input capacitances are dominant which favors relatively large power reductions through gate sizing. The effectiveness of DSVS does not depend on the said capacitance ratio, as can be seen from Equation 5.5. This perception motivated an analysis of future scenarios that has been published in [77]. Using a simple node capacitance model and previously published device and interconnect scaling roadmaps, the ratio of interconnect capacitances to device capacitances is predicted to increase in future technology generations. If this projection comes true, gate sizing will become less effective in favor of DSVS.

The above capacitance analysis is based solely on the information available in the synthesis library. Thus, the interconnect capacitance values are only rough estimates. However, these are the parameters that actually drive the logic synthesis process and, hence, determine the quality of the synthesis results.

Strictly speaking, gate sizing not only reduces the gate input capacitances C_G but also the diffusion capacitances C_{DB}. However, the latter are typically not explicitly modeled in synthesis libraries and, hence, do not affect the synthesis results. For this reason, the above analysis has been restricted to the gate input and interconnect capacitances.

7.5.3.2 Slack Analysis

A comparison of the data in column three of Table 7.4 and column four of Table 7.3 reveals some correlation of the power reduction due to DSVS with the number of cells operated at V_{DDL}. If the number of low voltage cells is large, the power reduction tends to be large as well (e.g. i10, rot and x3), while a small number of low voltage cells usually yields little power savings (e.g. alu2 and vda). However, the power reduction is not strictly proportional to the number of low voltage cells, since different cells often make quite different contributions to the total dynamic power consumption (e.g. my_adder and rot). Sometimes even power reductions can be observed although the number of low voltage cells is zero (e.g. c499). This is due to some logic restructuring taking place during the DSV power optimization. A question that cannot be answered on the basis of this data is the question of which parameters determine the number of low voltage cells.

Since the idea of DSVS is to trade off slack against dynamic power, a possible measure of the optimization potential is the amount of available slack. Figure 7.2a shows the results of a slack distribution analysis of all 28 combinational benchmark circuits before power optimization (see also [78]). The analysis procedure that has been used is as follows. For every gate in every netlist, the longest path running through that gate is determined using static timing analysis. The slack of this longest path is then assigned to the respective gate. In the remainder of this document, this analysis procedure is referred to as the type-1 slack analysis. In Figure 7.2a, the slack distribution is depicted in the form of a histogram with

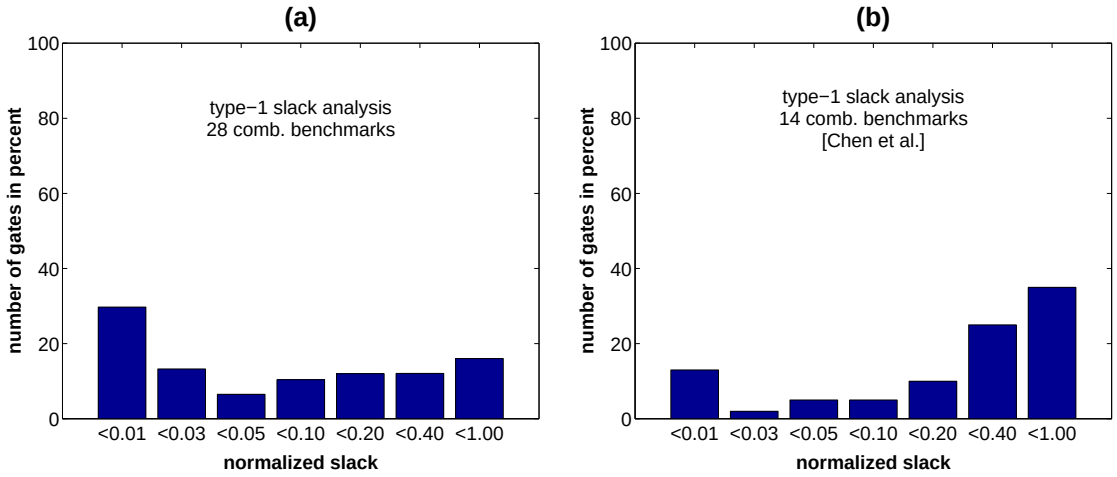

Figure 7.2: Type-1 slack distribution analysis after timing-driven synthesis: (a) results for 28 benchmark circuits; (b) results for 14 benchmark circuits reproduced from [20].

the percentage of cells (gates) on the vertical axis and the slack normalized to the timing constraint, i.e. the largest acceptable critical path delay t_{max}, on the horizontal axis. The horizontal axis is divided into seven intervals with increasing width from left to right (from smaller to larger slack). The normalized slack values contained in the figure denote the upper limits of the intervals. The height of the bars is proportional to the number of gates that have been assigned a slack value from the respective slack interval.

A similar analysis was carried out by Chen et al. on a selection of 16 combinational benchmark circuits after timing-driven synthesis subject to the strictest timing constraints [20]. From the results, which are reproduced in Figure 7.2b, Chen at al. concluded that there was a large potential for power reduction using DSVS because of the large number of non-critical cells. From a comparison of the two bar graphs, however, it is evident that, in the experiments discussed in this chapter, the benchmark circuits have been more critical. Moreover, the average normalized slack, a parameter used by Chen et al. for quantifying the optimization potential, has been 0.164, as opposed to 0.354 in the work by Chen et al. Consequently, there has been less potential for power-delay-trade-off. Since the timing constraints used by Chen et al. were even more strict than in this work, this discrepancy must be accredited to the capabilities of the tools used for timing-driven synthesis (see Section 7.5.2), rather than the constraints.

As mentioned in Section 5.3, Usami et al. also published slack distributions for the modules to which they applied DSVS. Although the analysis procedure was slightly different than that used here – they counted the number of paths with a certain amount of slack instead of the number of gates – the results show that the timing constraints were far from being strict. Although Yeh et al. did not carry out a slack distribution analysis, the following three arguments suggest that the optimization potential was similar to, if not larger than, that in

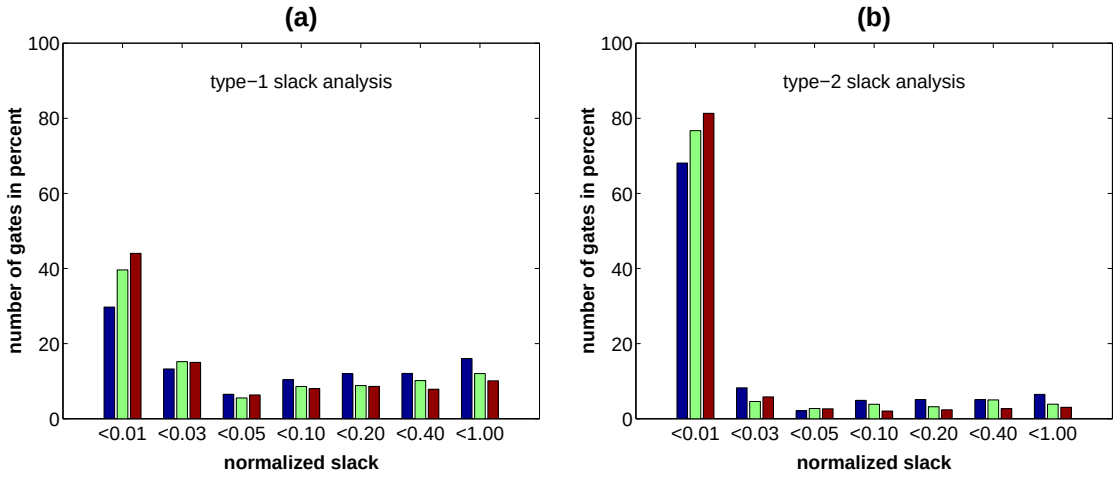

Figure 7.3: Slack statistics for 28 combinational benchmarks: (a) results of type-1 slack analysis and (b) results of type-2 slack analysis before power optimization (left bar), after SSV power optimization (middle bar), and after DSV power optimization (right bar).

the work by Chen et al. Firstly, both groups used the same tool (SIS) for timing-driven synthesis. Secondly, as opposed to Chen et al., Yeh et al. relaxed the timing constraints somewhat. Finally, both groups obtained similar results when they used CVS as a reference in their design environments.

The above discussion of slack distributions reveals one reason for DSVS being less effective than promised by other researchers when used in a state-of-the-art design environment: the use of commercial synthesis tools and realistic timing constraints results in relatively small amounts of slack. Another important aspect is the extensive use of SSV power optimization, particularly gate sizing, as part of the DSV logic synthesis methodology in this study. The discussion at the beginning of this section has already shown that DSVS is primarily used for exploiting slack that has been left unutilized by gate sizing. Hence, a slack distribution analysis carried out after SSV power optimization gives a better impression of the optimization potential left for DSVS.

In Figure 7.3a, there are three bars associated with each slack interval. In each group of three bars, the left bar corresponds to the situation after timing-driven synthesis, the middle bar represents the results of SSV power optimization, and the right bar describes the situation after DSV power optimization. From the graph, it can be ascertained that SSV power optimization has significantly increased the number of critical cells and, hence, reduced the optimization potential. As a result, the increase in the number of critical cells during DSV power optimization has been comparatively small.

It should be noted that in the type-1 slack analysis the restrictions arising from the level-conversion issue are ignored. As discussed in Section 5.1, the supply voltage of non-critical gates cannot be reduced if there is at least one critical cell in the fan-out which must be

operated at the higher supply voltage. In order to take this into account, the slack analysis can be modified as follows. For every gate in the netlist, type-1 slack analysis is recursively used on all gates in its fan-out. The slack of the most critical fan-out gate is then assigned to the gate under consideration. In other words, every gate in the netlist is considered equally critical as the most critical gate in its fan-out. This analysis procedure is named type-2 slack analysis in this study. Figure 7.3b shows the results of type-2 slack analysis for the 28 combinational benchmark circuits. From this graph, it is even more evident that supply voltage scaling can be applied only to a small number of cells.

7.5.3.3 Prediction of Potential Power Savings

The two different types of slack distribution analyses discussed in the previous paragraph can give only a first impression of how critical a circuit is. The slack of a cell is a local criterion which is not sufficient to decide whether voltage scaling is actually applicable to that cell. A quantitative prediction of the power savings requires a more complex analysis procedure that evaluates not only the slack but also the actual delay increment that must be expected to result from a reduction of the supply voltage of a cell and all its fan-out cells. For this purpose, the PSEM introduced in Section 6.3 has been developed here.

In Figure 7.4a, the actual power reductions achieved through DSVS as a part of the DSV logic synthesis methodology (filled circles) and through CVS (diamonds) on the 28 combinational benchmark circuits are compared with the values calculated from Equation 6.12 (solid line). The PSEM has been used for determining the power saving index *PSX* for each individual circuit. Three important conclusions can be drawn from the figure. Firstly, the existence of a correlation between the parameter *PSX* and the actual power savings is confirmed. Secondly, the results of CVS track the prediction very well. This results from the restriction of both algorithms to DSVS as a standalone optimization method, which means that the structure of the logic is invariant. Thirdly, the DSV logic synthesis methodology typically yields better results than predicted. This can be accredited to the use of DSVS in combination with a number of logic transformations (see Chapter 3). These transformations can help improving the optimization potential for DSVS while the optimization is in progress. The mechanisms implemented in the PSEM cannot predict the effect of these transformations, which explains the relatively large deviation of the actual power savings from the linear interpolation, i.e. from the dashed line in Figure 7.4a.

The PSEM can be applied to an existing SSV gate-level netlist without the need for a DSV library. The only requirements are the availability of a static timing analyzer, a power analysis tool, and a conventional SSV library that has been timing- and power-characterized at the nominal supply voltage V_{DD}. This facilitates a prediction of the effectiveness of DSVS for specific modules before spending the effort of developing a DSV library. The current implementation of the PSEM, however, works only on combinational circuits.

Figure 7.4b shows the predicted average power reduction for the 28 combinational benchmark circuits as a function of V_{DDL}. This analysis has been conducted using the PSEM.

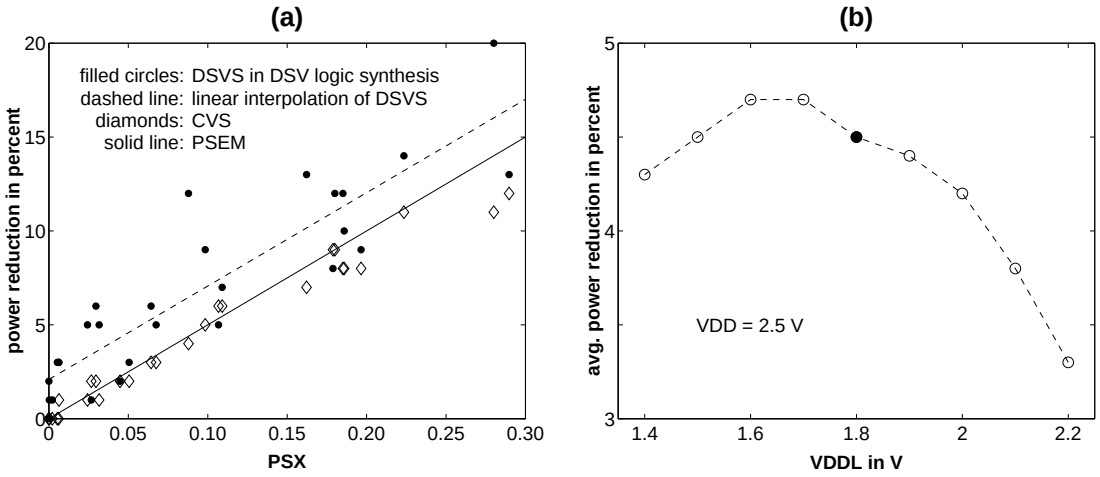

Figure 7.4: Application of the power savings estimation method (PSEM): (a) quantitative analysis of the optimization potential of individual circuits and comparison with results of DSV logic synthesis and CVS; (b) estimated average power reduction for 28 combinational benchmark circuits as a function of V_{DDL}.

Obviously, the optimal V_{DDL} can be expected between 1.5 V and 1.8 V, where the dependence of the power reduction on V_{DDL} is relatively weak. The analysis confirms that the value of 1.8 V, which has been used in this study, is reasonably close to the optimum. Moreover, the analysis illustrates that the PSEM could also help in finding the optimal choice of V_{DDL} prior to the characterization of the DSV standard cell library.

7.5.4 Consequences of Varying Delay Constraint Strictness

In Section 7.5.1, the optimization of numerous benchmark circuits under relaxed timing constraints has been discussed. In those experiments, the timing constraints have been moderately relaxed to 1.2 times the shortest possible critical path delays t_{cmin}. In another series of experiments, a subset of ten combinational benchmarks, namely the ISCAS'85 benchmarks, has been optimized under varying timing constraint strictness, i.e. the critical path delays have been set to 1.1, 1.35, and 1.5 times the minimum (t_{cmin}). The results can be seen in Figure 7.5a. Note that in this figure, just as in Table 7.3, the results of SSV and DSV power optimization are compared with the power consumption before power optimization, while the results of DSVS are compared with the power consumption after SSV power optimization. The figure contains average values for the set of ten circuits. Results for individual circuits can be found in Tables B.3, B.4, and B.5 in the appendix and in [76].

The power reduction due to DSV power optimization increases as the timing constraints are relaxed (see uppermost curve in Figure 7.5a). Starting from relatively strict constraints, the power reduction increases at a high rate first, and, the more the constraints are relaxed,

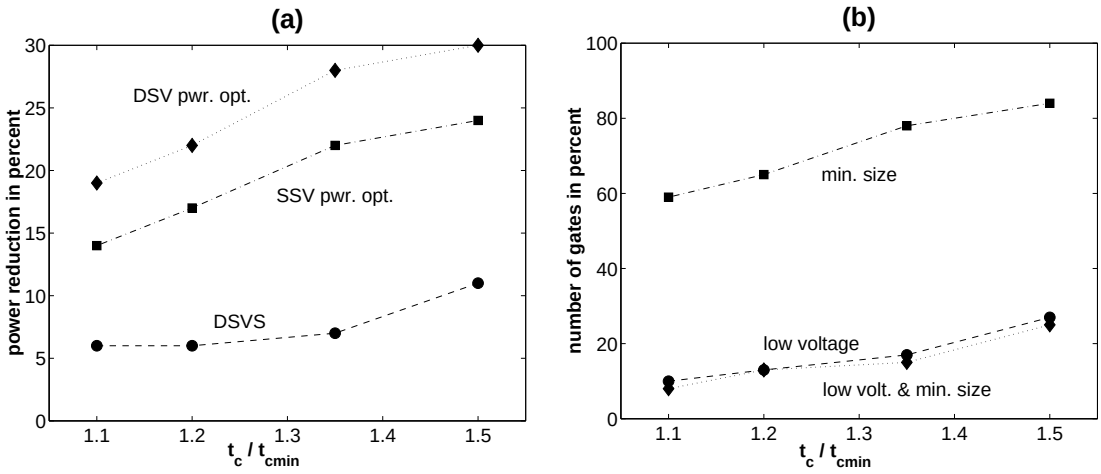

Figure 7.5: Effectiveness of SSV optimization, DSV logic synthesis (incl. SSV techniques), and DSVS as a function of critical path delay for ISCAS'85 benchmarks: (a) power reduction as a function of the normalized critical path delay; (b) impact of the delay on the number of minimum size and low voltage cells in the final DSV implementations.

the smaller becomes the rate at which the power reduction increases. This characteristic is primarily determined by the similar characteristic of the SSV power optimization, which is an integral part of the DSV optimization (see middle curve). This behavior can be explained as follows. If the constraints are relaxed starting from a relatively strict level, a large portion of the additional slack created thereby can be exploited for power reduction through gate sizing. If the constraints are relaxed starting from a less strict level, many cells have minimum size already before the relaxation. Consequently, a smaller portion of the additional slack created can be exploited by gate sizing. This can be seen from Figure 7.5b, where the number of minimum size cells increases at a declining rate as the timing constraints are relaxed. The numbers of minimum size, low voltage and minimum size low voltage cells for individual circuits can be found in Table B.6 in the appendix.

The opposite behavior can be observed in the case of DSVS (see lower curve in Figure 7.5a). The smaller the portion of the additional slack exploited by gate sizing, the more slack is left to be utilized by DSVS. Therefore, the power reduction increases at an increasing rate as the timing constraints are relaxed. The lower curves in Figure 7.5b show that the number of low voltage cells actually increases at an increasing rate as the timing constraints are relaxed. These two curves also confirm what has been said in the first paragraphs of Section 7.5.3 about the effectiveness of the two techniques: almost all low voltage cells have minimum size, which implies that DSVS is used primarily if no further improvement can be achieved through gate sizing.

7.5.5 Comparison with Global Supply Voltage Scaling

Circuits that are not subject to the strictest timing constraints can be optimized by means of power-driven SSV or DSV logic synthesis under the actual relaxed timing constraints for operation at the nominal supply voltage V_{DD}, as discussed in Sections 7.5.1 and 7.5.4. Alternatively, the GSVS approach explained in Sections 4.2 and 7.4 can be used for dynamic power optimization.

Scaling the supply voltage globally from the nominal value V_{DD} down to a lower value V_{DDp} increases the delay by a factor of $1+p$. With Equation 2.15, this can be expressed as

$$\frac{\Delta t_D}{t_D} = 1 - \frac{V_{DDp}}{V_{DD}} \left(\frac{V_{DD} - V_t}{V_{DDp} - V_t} \right)^{\alpha} = p \quad . \tag{7.1}$$

Equation 5.10 with *psc* set to zero leads to the following conservative estimate of the power reduction achievable through GSVS:

$$\left| \frac{\Delta P_{dyn}}{P_{dyn}} \right| = 1 - \left(\frac{V_{DDp}}{V_{DD}} \right)^2 \tag{7.2}$$

In a number of experiments, the delay and the power consumption of the ISCAS'85 benchmark circuits have been analyzed for global supply voltages of 2.5 V and 1.8 V using the high and low voltage subsets of cells from the DSVL025 library. It has been observed that the critical path delays have been about 35% larger at the lower voltage, i.e. p is equal to 0.35. With this result and V_t equal to 0.5 V, it follows from Equation 7.1 that α is 1.46 for this technology. The dynamic power consumption of the circuits has been about 50% lower at 1.8 V, which agrees well with the estimate obtained from Equation 7.2.

The amount of global voltage reduction is maximized if the fastest possible implementation of a circuit is used as the starting point for GSVS. Therefore, in a first set of experiments, the ISCAS'85 benchmark circuits had been synthesized and SSV power optimized under the strictest timing constraints before the supply voltage has been scaled down to the minimum. This strategy is named GSVS (I) in this study.

In Figure 7.6, the results of GSVS (I) and DSVS are depicted. Timing constraints of 1.2, 1.35, and 1.5 times the shortest possible critical path delays have been chosen for this comparison. This corresponds to p equal to 0.2, 0.35, and 0.5, respectively. For p equal to 0.2 and 0.5, the target voltages V_{DDp} have been calculated from Equation 7.1 and the expected power consumption after GSVS has been derived from the power consumption of the SSV power optimized fastest implementations of the circuits using Equation 7.2. For p equal to 0.35, the target voltage V_{DDp} is 1.8 V, as mentioned above, and GSVS has actually been carried out using the low voltage subset of cells from the DSVL025 synthesis library. The cell area associated with GSVS is always the same as that of the SSV power optimized fastest implementations. The power consumption and the cell area of the circuits after SSV

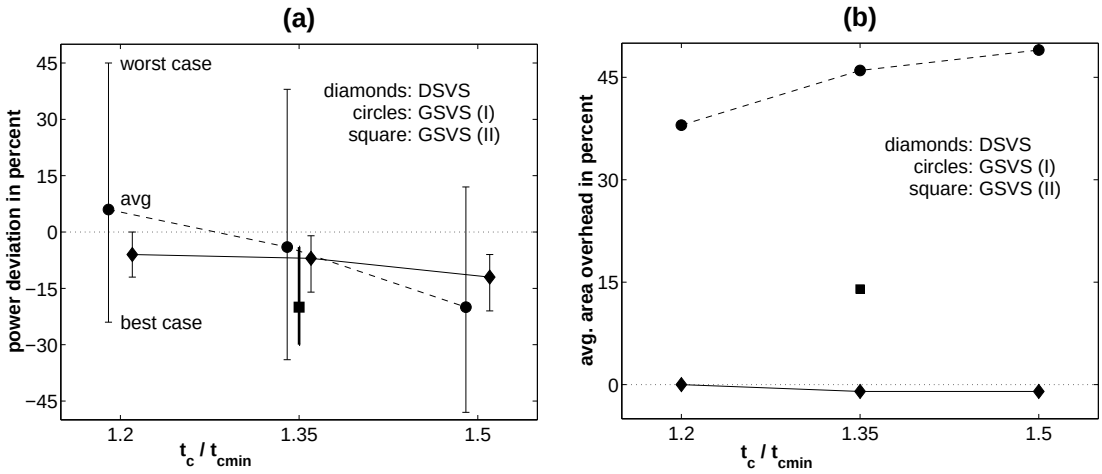

Figure 7.6: Comparison of DSVS with two different GSVS strategies for ISCAS'85 benchmarks: (a) power and (b) area of DSVS- and GSVS-optimized circuits compared with the results of SSV power optimization under relaxed constraints.

power optimization under relaxed constraints using the nominal supply voltage of 2.5 V have been used as reference values in this comparison. The figure shows average values for all ISCAS'85 benchmark circuits. In addition, in Figure 7.6a, the ranges of values from the best to the worst cases are marked by means of error bars. The detailed results for individual circuits can be found in Tables B.7, B.8 and B.9 in the appendix and in [76].

For a critical path delay of 1.2 times the shortest possible delay, DSVS has led to an average of 6% lower power consumption than SSV power optimization. For GSVS, a target supply voltage of 2.0 V has been determined. Under these circumstances, GSVS is expected to result in 6% higher power than SSV power optimization. Also, the effectiveness of GSVS appears to be extremely circuit dependent. In the worst case, the power consumption has been up to 45% larger after GSVS than after SSV power optimization. In the best case, the power has been 24% lower after GSVS. As shown in Figure 7.6b, the cell area is generally much larger for GSVS optimized circuits; the area penalty has been 38% on average.

In the second case, where the critical path delays have been relaxed to 1.35 times the shortest possible delay, GSVS with a target value of 1.8 V has led to 4% lower power on average. Obviously, the average effectiveness of GSVS has increased compared with the previous case, where the timing has been more strict. However, it is still lower than that of DSVS, which has yielded 7% power reduction on average. Also, there is still a large variation of the individual results. In the worst case, the power consumption has been 38% larger, while, in the best case, it has been 34% lower after GSVS. The area penalty has increased to 46% on average.

In the third case, i.e. for critical path delays 1.5 times larger than the minimum delays, a minimum supply voltage of 1.6 V has been determined for GSVS. The average power re-

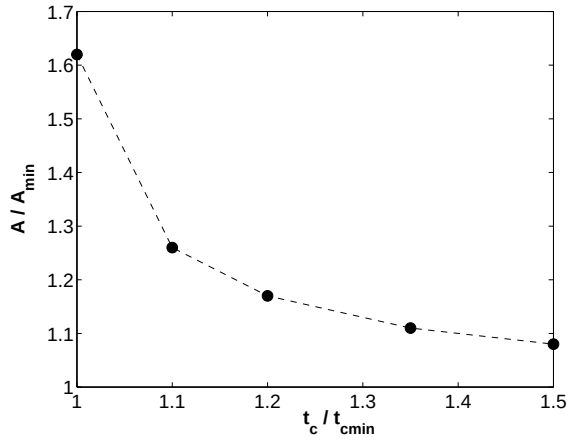

Figure 7.7: Average normalized area as a function of normalized delay for ISCAS'85 benchmark circuits.

duction due to GSVS has been 20% as opposed to 12% due to DSVS. The tremendous variation of the individual results can still be observed and the area penalty has again slightly increased compared with the second case.

These results show that the effect of GSVS on individual circuits is less predictable than that of DSVS. For relatively strict constraints, the GSVS (I) strategy is inferior to DSVS (and even to SSV power optimization) regarding both the power consumption and the area. The effectiveness of both GSVS and DSVS increase as the timing constraints are relaxed, with the effectiveness of GSVS increasing at a higher rate. Thus, regarding the average power reduction, GSVS is advantageous in the case of extremely relaxed timing constraints. For DSVS, at a fixed supply voltage, less strict timing constraints result in smaller circuits. The area of the circuits optimized using the GSVS strategy, however, is always equal to the area of the fastest possible implementation. Therefore, GSVS results in larger circuits and the area penalty increases as the timing constraints are relaxed.

A closer look at the relation between cell area and delay implies an improved GSVS strategy. As can be seen in Figure 7.7, the fastest possible implementations of the ISCAS'85 benchmark circuits are about 60% larger than the smallest possible implementations. This is due to massive logic parallelization and gate up-sizing occurring during logic synthesis subject to the strictest constraints. As the timing constraints are relaxed, the total cell area A decays rapidly first and then slowly converges to its minimum A_{min}. The large area overhead that can be observed in the case of the strictest timing constraints compared with slightly relaxed constraints brings about a power overhead that largely detracts from the effectiveness of GSVS. This implies that somewhat relaxing the timing constraints during the initial timing-driven synthesis and the SSV power optimization might improve the results obtained from GSVS. This strategy is denoted GSVS (II) in this study

The above argument is confirmed by the results of GSVS (II) that are included in Figure 7.6.

Gate type:		INV	BUF	NAND	AND	NOR	OR	XNOR	XOR
$\Delta t_D/t_D$	min.	30%	30%	30%	30%	0%	0%	0%	10%
	max.	60%	60%	70%	50%	60%	50%	60%	60%
	avg.	50%	40%	50%	40%	30%	30%	40%	40%

Table 7.5: Relative gate delay increment due to aggravated body effect in the dual power rail layout scheme. The minimum and maximum values have been observed at different corners of the characterization parameter space, i.e. different values for t_T and C_{node}.

In these experiments, GSVS has been applied to circuits initially optimized for critical path delays of 1.1 times the shortest possible delays. The actual timing constraints for the final implementations have been set to 1.35 times the shortest possible delays. The detailed results can be found in Table B.10 in the appendix. The advantage of this strategy is obvious. Although the supply voltage has been reduced only to 2.0 V as opposed to 1.8 V for GSVS (I), the average power reduction compared with SSV power optimization under the actual relaxed constraints has increased from 4% to 20% and there is much less variation. In the worst case, the power reduction has been 4% and, in the best case, 30% has been observed. This strategy makes GSVS superior to DSVS for a wider range of timing constraints; the reason is the smaller area overhead of only 14% as opposed to 46% using GSVS (I).

7.5.6 Impact of Layout Concepts on Logic Synthesis

Low voltage cells suffer from performance degradation due to aggravated body effect, when the n-wells of both low and high voltage cells are tied to the same potential. This problem arises in the dual power rail layout scheme, as explained in Section 5.4. In order to investigate the possible impact of this effect on the results of DSV logic synthesis, the DSVL025 synthesis library has been re-characterized with the bulk terminals of the p-channel transistors in the low voltage cells connected to V_{DD} instead of V_{DDL}. The numbers included in Table 7.5 show that the delay of different types of gates from the DSVL025 library increases by up to 70% and 30% to 50% can be considered typical. This extra delay must be expected to reduce the number of low voltage cells, thus detracting from the possible power savings.

The modified synthesis library has been used for optimizing the complete set of 28 combinational benchmark circuits again. The results of these experiments – details can be found in Table B.11 in the appendix – show that the number of cells operated at V_{DDL} has actually decreased significantly under the circumstances explained above. On average, only 15% of all cells are operated at the lower voltage as opposed to 21% in the case without the aggravated body effect. The power savings have also decreased but the effect is less

notable. This is most likely due to a reduction of the short-circuit currents due to increased threshold voltages, which compensates partly for the loss of capacitive power savings. The average power reduction due to DSVS has been 6% as opposed to 7% in the case without the aggravated body effect.

Although the degradation of the effectiveness of DSVS is less significant than expected, it is large enough to eliminate the advantage of the smaller power overhead in the dual power rail layout scheme compared with the split-row layout style. Consequently, according to the results published by Yeh et al. (see Section 5.4), a power overhead of as much as 5% of the power consumption of a corresponding SSV implementation or 20% of the power savings achieved in the pre-layout design phase must be expected. Clearly, this overhead is not negligible, particularly in view of the true effectiveness of DSV logic synthesis discussed in the preceding sections.

7.6 Optimization of Sequential Circuits

7.6.1 Single and Dual Supply Voltage Power Optimization

The set of 20 sequential MCNC benchmark circuits (see Table 7.2) has been optimized under the strictest delay constraints. The results are summarized in Tables 7.6 and 7.7.

The second column of Table 7.6 shows the power reduction that has been achieved using only state-of-the-art SSV power optimization techniques. Compared with the results of timing-driven logic synthesis, the dynamic power consumption has been reduced by 7% on average and by 12% at the most (mm30a, mult16a, s1196). According to the numbers in the third column, the average power reduction has increased to 11% due to DSVS and SSV power optimization being used simultaneously. In the best case, DSVS has increased the total power reduction from 4% to 21% (s38417). In the fourth column, the dynamic power consumption after DSV optimization is compared with the dynamic power after SSV power optimization, i.e. the additional benefit of DSVS is shown. The average power reduction due to DSVS has been 4%. In the best case, the improvement has been 17% (s38417), and in the worst case DSVS has not led to any additional power reduction at all (mult16a, s1488). In three out of 20 cases, significant power savings of more than 10% have been achieved (s820, s832, s38417) through DSVS.

In the rightmost column, the total cell area after DSV optimization is compared with the total cell area after SSV optimization. On average, the area has increased by 3%. This is due to the larger area of the level-converting flip-flops. The largest area overhead has been observed for s9234.1 and s38417 where, firstly, a large number of level-converting flip-flops has been used (see rightmost column of Table 7.7) and, secondly, a large portion of all cells have been flip-flops (see Table 7.2).

	P_{dyn} after			Cell area
	SSV pwr. opt.	DSV$^{(*)}$ pwr. opt.	DSVS	
	comp. with before pwr. opt.		comp. with after SSV pwr. opt.	
bigkey	-3%	-4%	-1%	-1%
clma	-1%	-2%	-1%	±0%
mm4a	-10%	-12%	-2%	+10%
mm30a	-12%	-18%	-6%	+10%
mult16a	-12%	-12%	±0%	±0%
s344	-8%	-10%	-2%	-1%
s349	-1%	-6%	-5%	+1%
s382	-5%	-8%	-3%	±0%
s444	-9%	-11%	-2%	±0%
s526	-3%	-4%	-1%	±0%
s641	-7%	-12%	-6%	+4%
s713	-11%	-13%	-3%	+1%
s820	-2%	-12%	-10%	-5%
s832	-8%	-20%	-14%	-6%
s1196	-12%	-17%	-5%	+6%
s1488	-8%	-8%	±0%	±0%
s1494	-11%	-11%	-1%	±0%
s9234.1	-6%	-14%	-8%	+15%
s38417	-4%	-21%	-17%	+28%
sbc	-11%	-12%	-1%	±0%
avg.	-7%	-11%	-4%	+3%

Table 7.6: Optimization of sequential benchmarks subject to the strictest timing constraints. $^{(*)}$DSV power optimization includes both DSVS and SSV techniques.

The data in columns two to three of Table 7.7 confirms what has been observed already in the case of combinational benchmark circuits: gate sizing is generally more effective under the given circumstances and has been preferred to DSVS. The latter technique has been used primarily for exploiting slack that could not be utilized by gate sizing.

Both types of slack distribution analysis have been carried out before and after SSV and DSV power optimization. A comparison of the diagrams in Figures 7.3 and 7.8 reveals that there has existed a significantly smaller amount of slack in the sequential benchmark cir-

	Amount of gates with			Amount of FF with	
	min. size	low voltage	both	low voltage	lev. conv.
bigkey	22%	$< 1\%$	$< 1\%$	0	0
clma	12%	$< 1\%$	$< 1\%$	0	27%
mm4a	47%	1%	1%	0	50%
mm30a	43%	9%	7%	11%	68%
mult16a	63%	1%	1%	0	0
s344	26%	1%	1%	0	0
s349	36%	4%	2%	0	13%
s382	27%	2%	1%	5%	5%
s444	45%	2%	1%	10%	0
s526	30%	0	0	0	0
s641	42%	16%	15%	5%	16%
s713	48%	11%	10%	11%	5%
s820	31%	11%	10%	0	0
s832	42%	10%	10%	0	0
s1196	56%	13%	12%	6%	50%
s1488	31%	7%	6%	0	0
s1494	34%	2%	2%	0	0
s9234.1	31%	11%	10%	5%	30%
s38417	34%	12%	9%	17%	70%
sbc	52%	5%	5%	0	0
avg.	38%	6%	5%	3%	17%

Table 7.7: Properties of sequential benchmarks after DSV power optimization under the strictest timing constraints.

cuits that have been subject to the strictest timing constraints. Moreover, for the non-power-optimized sequential circuits, an average normalized slack of 0.107 has been determined by means of type-1 slack analysis; the corresponding value for the non-power-optimized combinational circuits is 0.164. This explains why both SSV and DSV power optimization have been less effective in this case.

The same circuits have been optimized under moderately relaxed timing constraints, i.e. the critical path delays have been constrained to 1.2 times the shortest possible delays. This has created additional slack, as the diagrams in Figure 7.9 show. An average normalized slack

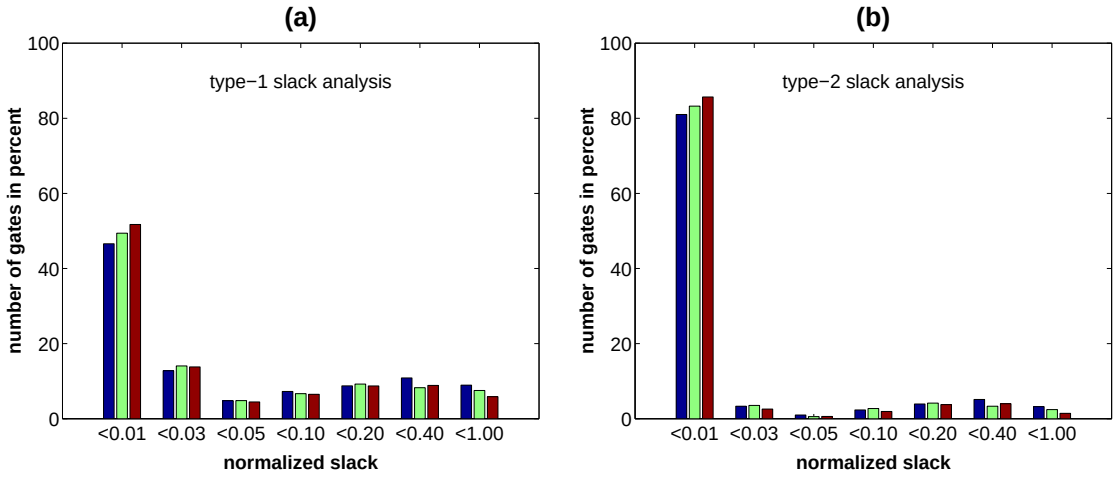

Figure 7.8: Slack statistics for 20 sequential benchmarks subject to the strictest constraints: (a) results of type-1 slack analysis and (b) results of type-2 slack analysis before power optimization (left bar), after SSV power optimization (middle bar), and after DSV power optimization (right bar).

of 0.179, which is close to the slack of the combinational circuits subject to the same timing constraint strictness, has been determined by means of type-1 slack analysis before power optimization. The larger slack has enabled larger power savings due to SSV and DSV power optimization, as can be seen from the results presented in Table 7.8. The total power reduction achieved through DSV power optimization has increased significantly in every single case. The average value has increased from 11% to 23%. The benefit of DSVS has increased from 4% to 8% on average. Power savings of more than 10% have been observed in seven out of 20 test cases. For a few individual circuits, the benefit of DSVS has been smaller under moderately relaxed constraints than in the case of more strict constraints (s349, s820, s832). In these cases, the SSV optimization techniques have made even larger contributions to the total power savings.

The cell area overhead has been 8% on average as opposed to 3% for the strictest constraints. This increase is the result of a larger number of level-converting flip-flops, as indicated by the numbers in the rightmost column of Table 7.9. The other data included in this table prove that the larger slack has actually allowed more cells to be scaled to minimum size and more cells to be operated at the lower supply voltage which explains the increased effectiveness of both SSV power optimization and DSVS. Again, almost all low voltage cells have minimum size, which indicates that DSVS has been used where gate sizing has left slack unutilized.

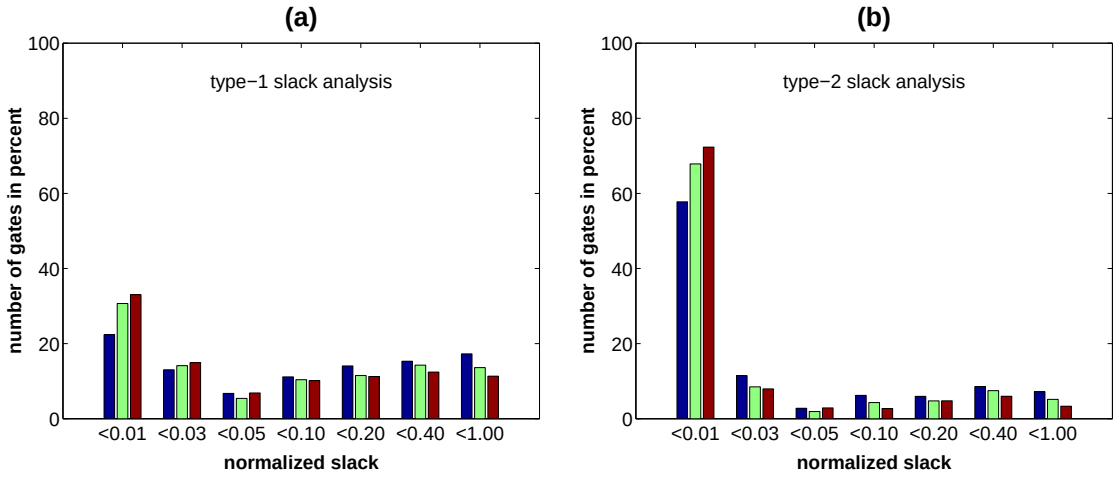

Figure 7.9: Slack statistics for 20 sequential benchmarks subject to moderately relaxed constraints: (a) results of type-1 slack analysis and (b) results of type-2 slack analysis before power optimization (left bar), after SSV power optimization (middle bar), and after DSV power optimization (right bar). Critical path delays set to 1.2 times the minimum.

7.6.2 Feasibility of Clock Voltage Scaling

The sequential benchmarks have been synthesized and optimized again under moderately relaxed timing constraints, i.e. the critical path delays have been constrained to 1.2 times the shortest possible delays. This time, the designs have been prepared for clock voltage scaling by disabling high voltage flip-flops throughout the entire design process from the initial timing-driven synthesis to the final DSV power optimization.

The level-converting flip-flops used in these experiments have larger clock-to-output delays and larger setup times (see Section 6.5.2) and, hence, introduce extra delay into all paths. Because of this extra delay, it has been impossible to meet the timing constraints in nine out of 20 test cases. Some characteristics of the eleven remaining circuits are summarized in Tables 7.10 and 7.11.

Columns two to four of Table 7.10 contain parameters that appear in Equation 5.12 and, hence, determine the overall effectiveness of clock voltage scaling. These parameters are the relative contributions of the clock network (P_{clk}), the combinational parts of the circuits (P_{comb}) and the sequential elements (P_{seq}) to the total dynamic power consumption P_{dyn}. The amount of minimum size cells in relation to the total number of cells is given in the fifth column. This is regarded as a measure of the degree to which gate sizing has been used for power reduction. The numbers in the sixth column describe how many level-converting flip-flops each circuit contains and the data in the rightmost column shows the area A_{seq} occupied by sequential elements in relation to the total cell area A. This information helps in explaining the area overhead caused by clock voltage scaling.

	P_{dyn} after			Cell area
	SSV pwr. opt.	DSV$^{(*)}$ pwr. opt.	DSVS	
	comp. with before pwr. opt.		comp. with after SSV pwr. opt.	
bigkey	-17%	-20%	-4%	+2%
clma	-21%	-25%	-5%	+2%
mm4a	-18%	-25%	-8%	+14%
mm30a	-22%	-35%	-17%	+16%
mult16a	-15%	-23%	-10%	+14%
s344	-10%	-22%	-13%	+13%
s349	-15%	-17%	-2%	+2%
s382	-8%	-13%	-6%	+7%
s444	-10%	-22%	-13%	+7%
s526	-12%	-15%	-4%	+3%
s641	-13%	-19%	-7%	+9%
s713	-16%	-22%	-8%	+9%
s820	-18%	-21%	-5%	+1%
s832	-22%	-23%	-2%	$\pm$0%
s1196	-24%	-33%	-12%	+9%
s1488	-25%	-29%	-6%	$\pm$0%
s1494	-24%	-27%	-4%	$\pm$0%
s9234.1	-6%	-20%	-15%	+25%
s38417	-5%	-22%	-18%	+28%
sbc	-18%	-23%	-6%	+5%
avg.	-16%	-23%	-8%	+8%

Table 7.8: Optimization of sequential benchmarks under moderately relaxed timing constraints. Critical path delays set to 1.2 times the minimum. $^{(*)}$DSV power optimization includes both DSVS and SSV techniques.

Table 7.11 describes the effect of clock voltage scaling on the power consumption and the cell area. Columns two to four show the changes in the total dynamic power, the power consumed by the combinational parts of the circuits and the power consumed by the sequential elements. These parameters also appear in Equation 5.12 and, hence, determine the effectiveness of clock voltage scaling. The next two columns describe to which degree the additional delay introduced by the level-converting flip-flops has increased the complexity of the circuits and reduced the number of minimum size gates. This data explains

	Amount of gates with			Amount of FF with	
	min. size	low voltage	both	low voltage	lev. conv.
bigkey	72%	1%	1%	0	0
clma	35%	2%	2%	0	64%
mm4a	74%	11%	11%	8%	50%
mm30a	72%	35%	32%	32%	58%
mult16a	78%	19%	19%	0	50%
s344	71%	4%	4%	0	27%
s349	73%	5%	4%	0	7%
s382	71%	6%	6%	5%	14%
s444	80%	17%	12%	19%	14%
s526	77%	2%	2%	0	10%
s641	63%	18%	17%	16%	26%
s713	76%	22%	20%	21%	26%
s820	67%	10%	10%	0	0
s832	72%	7%	7%	0	0
s1196	79%	38%	36%	22%	56%
s1488	71%	26%	23%	0	0
s1494	76%	15%	15%	0	0
s9234.1	44%	14%	13%	5%	55%
s38417	43%	14%	10%	19%	65%
sbc	69%	28%	24%	4%	26%
avg.	68%	15%	13%	6%	27%

Table 7.9: Properties of sequential benchmarks after DSV power optimization under moderately relaxed timing constraints.

the increase in the power consumed by the combinational parts of the circuits. Finally, the number of level-converting flip-flops and the total cell area overhead are shown in the two rightmost columns. All the values in this table, except for the number of level-converting flip-flops, have been calculated in relation to the characteristics of the DSV implementations with high voltage clocks. An important parameter that has not been included in this table is $\Delta P_{clk}/P_{clk}$, which describes the power savings in the clock network. This factor has been roughly 0.5 in all test cases which agrees with the expected value calculated from Equation 5.10.

	$\frac{P_{clk}}{P_{dyn}}$	$\frac{P_{comb}}{P_{dyn}}$	$\frac{P_{seq}}{P_{dyn}}$	Number of min. size gates	Number of lev.-conv. FF	$\frac{A_{seq}}{A}$
bigkey	6%	78%	16%	68%	0	22%
clma	5%	88%	7%	35%	64%	6%
mm4a	5%	77%	18%	69%	50%	34%
mm30a	4%	89%	7%	68%	58%	31%
mult16a	5%	74%	21%	72%	50%	33%
s641	8%	69%	23%	57%	26%	35%
s1196	3%	89%	8%	77%	56%	18%
s1488	2%	91%	7%	71%	0	4%
s1494	1%	93%	6%	75%	0	4%
s9234.1	28%	31%	41%	38%	55%	50%
s38417	36%	33%	31%	38%	65%	50%

Table 7.10: Characteristics of sequential benchmark circuits with high voltage clocks.

The effect of the clock voltage scaling technique on the total dynamic power consumption is very different for different circuits. In the two best cases (s9234.1, s38417), power savings of 21% to 27% have been achieved. Five other circuits (clma, mm30a, mult16a, s641, s1196) consume roughly the same power as they do without clock voltage scaling. In the four remaining test cases (bigkey, mm4a, s1488, s1494), clock voltage scaling has led to significant power overheads of up to 35%.

The first thing to notice is that the level-converting flip-flops have not directly created any power overhead. On the contrary, the dynamic power consumption of all sequential elements has generally been reduced, as the numbers in the fourth column of Table 7.11 show. The power overheads are solely due to restructuring of the combinational logic and gate up-sizing.

The different factors that determine the overall effectiveness of clock voltage scaling can be explained using Equation 5.12 and the data given in Tables 7.10 and 7.11. In the two best cases (s9234.1, s38417), the clock network and the sequential elements account for about 70% of the total dynamic power consumption. Therefore, the power reductions of 50% in the clock network and 10% to 20% in the sequential parts of the circuits predominate any power overhead in the combinational parts (10% for s38417). The result is large overall power savings. The area overhead is moderate in these two cases because the circuits contain many level-converting flip-flops even if clock voltage scaling is not used. Thus, the number of additional level-converting cells that have been inserted in order to facilitate clock voltage scaling is relatively small.

	$\frac{\Delta P_{dyn}}{P_{dyn}}$	$\frac{\Delta P_{comb}}{P_{comb}}$	$\frac{\Delta P_{seq}}{P_{seq}}$	Number of cells		Number of lev.-conv. FF	$\frac{\Delta A}{A}$
				total	min. size		
bigkey	+33%	+52%	-23%	+3%	-47%	100%	+43%
clma	±0%	+5%	-22%	+1%	-10%	100%	+4%
mm4a	+13%	+23%	-13%	+3%	-3%	92%	+15%
mm30a	±0%	+5%	±0%	-1%	+2%	72%	+3%
mult16a	+2%	+9%	-10%	+16%	+12%	100%	+22%
s641	-2%	+9%	-23%	+24%	-2%	79%	+35%
s1196	+2%	+4%	-10%	-4%	-15%	67%	+2%
s1488	+24%	+29%	-22%	+6%	-14%	100%	+20%
s1494	+35%	+40%	-22%	+8%	-36%	100%	+25%
s9234.1	-27%	-2%	-20%	+5%	-5%	89%	+16%
s38417	-21%	+10%	-10%	+1%	-5%	81%	+8%

Table 7.11: Characteristics of sequential benchmark circuits after clock voltage scaling. All numbers are in relation to the characteristics of the DSV implementations with high voltage clock. The only exception is the amount of level-converting flip-flops which is in relation to the total number of flip-flops.

In the two cases with the largest power overhead (bigkey, s1494), the sequential elements account for much smaller portions of the total dynamic power. Even more important are the small contributions of the clock networks of only 6% and 1%. This makes the power savings that have been achieved in this part of the circuits almost negligible. On the other hand, there are huge power overheads of 52% and 40% in the combinational parts of the circuits, which explains the large overall power overheads. These overheads are primarily due to gate up-sizing, as indicated by the significantly smaller amounts of minimum size gates. In both cases, many additional level-converting flip-flops have been inserted. This explains the tremendous area overhead observed for the circuit named bigkey. In the case of the circuit named s1494, the area overhead is somewhat smaller because the area occupied by the sequential elements is small compared with the total cell area. The overheads observed for mm4a and s1488 can be explained similarly.

In all other cases, the power savings in the sequential parts compensate for the relatively small overheads in the combinational parts, while the power savings in the clock networks are negligible. Thus, the overall effect of clock voltage scaling is negligible in these cases.

These experiments have shown which parameters determine the effectiveness of the clock voltage scaling approach. The important results are the following. Firstly, level-converting flip-flops are often more power efficient than their conventional counterparts. Secondly,

large power savings can be expected if primarily the clock network but also the sequential elements make large contributions to the total power consumption. Thirdly, the primary cause of power overheads is gate up-sizing in the combinational parts of the circuits. To a certain degree, the overheads are also due to more complex combinational logic. Finally, the area overhead depends on the portion of the total cell area occupied by sequential elements and on the number of additional level-converting flip-flops that have to be inserted in order to facilitate clock voltage scaling.

It should be noted again at this point that the analysis presented in this section covers only those aspects of clock voltage scaling that are visible in the pre-layout design phase. Other issues such as the clock skew and latency can only be addressed during the clock tree synthesis in the layout synthesis phase.

7.7 Comments

This is the first time that DSV power optimization at the logic level has been demonstrated to work in a conventional ASIC design environment. The results presented in this chapter prove the general feasibility of the proposed methodology and show that its effectiveness is superior to that of a previously published reference algorithm.

Because of the extensive use of state-of-the-art timing- and power-driven logic synthesis with the inevitable consequence of limited slack, the results do not feign an unrealistically large benefit of DSVS. In fact, this study reveals the fundamental limitations of DSVS at the logic level. The use of DSV logic synthesis should generally be restricted to circuits that are subject to the strictest or to moderately relaxed timing constraints. In the case of largely relaxed constraints, it is usually better to operate the entire circuit at the lower supply voltage. Since the average power savings that have been observed are below 10% and, according to the current standard of knowledge, the more complex DSV layout must be expected to cause a power overhead of up to 5% of the total dynamic power consumption or 20% of the pre-layout power savings, DSV logic synthesis should be further restricted to selected modules that have an optimization potential well above average.

An important characteristic of logic synthesis is the running time as a function of the complexity of the optimization problem. Important parameters determining the problem complexity are the size of the circuit to be synthesized and the number of cells in the library. When the proposed DSV logic synthesis methodology is used, the duration of a single power optimization run is usually longer than in state-of-the-art power-driven logic synthesis because of the increased number of cells in the library. In the experiments discussed in this chapter, up to 50% longer running times (24% on average) have been observed, which is considered a moderate penalty.

Chapter 8

Application to an Embedded Microcontroller

8.1 Digital Color Camera on a Chip

The LmDvp is a digital color image processor based on NATIONAL SEMICONDUCTOR'S 16-bit CompactRISC (CR16) processor. The chip has been designed to operate with NATIONAL SEMICONDUCTOR'S family of CMOS image arrays. In such a configuration, it provides a complete color camera solution for digital video and still imaging applications. A typical system configuration is depicted in Figure 8.1.

The highest level of the LmDvp design hierarchy is composed of four functional units: the image processing subsystem, the CompactRISC microcontroller subsystem, the video bus controller, and the system management unit (see Figure 8.2).

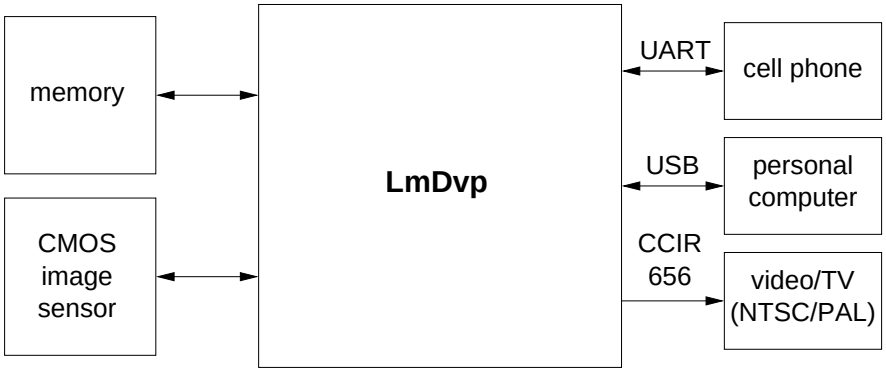

Figure 8.1: LmDvp in a color camera system environment.

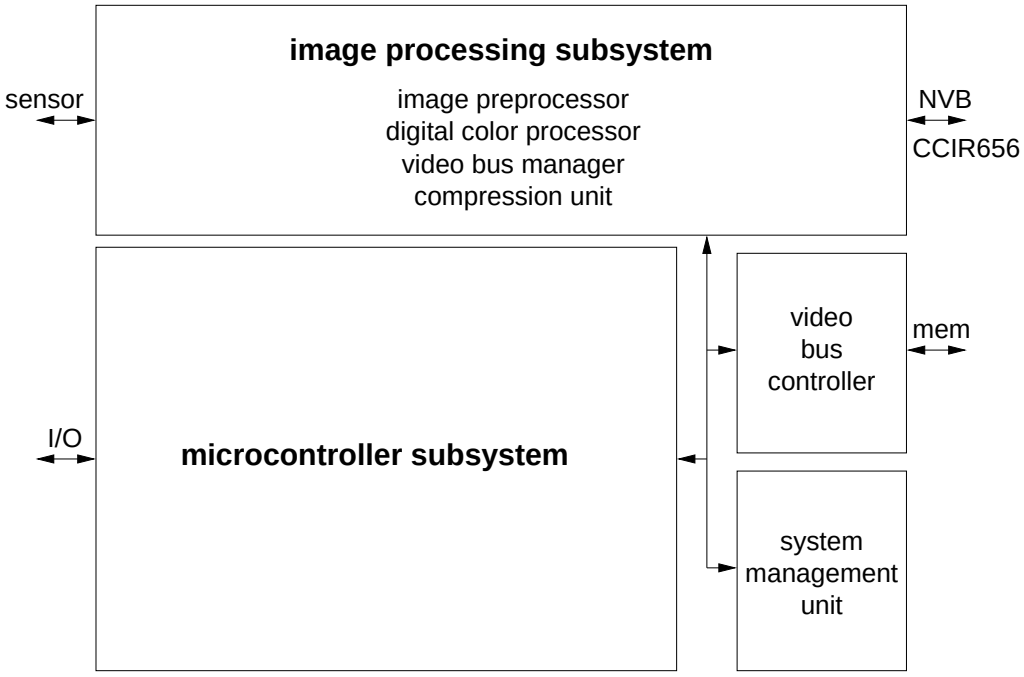

Figure 8.2: Simplified block diagram of the LmDvp chip.

The **image processing subsystem** consists of the following four submodules:

Image preprocessor In this module, the raw image data from the image sensor is captured and preprocessed.

Digital color processor This module is responsible for various types of color correction, color conversion, and similar operations.

Video bus manager This module provides an interface to NATIONAL SEMICONDUCTOR'S Video Bus (NVB) and a CCIR656[1] video output supporting NTSC (National Television Systems Committee) and PAL (Phase Alternation by Line) formats.

Compression unit In this unit, JPEG (Joint Photographic Experts Group) compression of video and still image data is carried out.

The CompactRISC **microcontroller subsystem** has been designed around the CR16 processor core and is responsible for house keeping, peripheral management, and so forth. It

[1] ITU-R Recommendation BT.656. Interfaces for digital component video signals in 525-line (e.g. NTSC) and 625-line (e.g. PAL) television systems. The Consultative Committee for International Radio (CCIR) was a predecessor organization of ITU-R.

accounts for about one half of the circuitry on the LmDvp chip. All the experiments discussed in this chapter have been carried out within this CR16 system environment. Therefore, the microcontroller subsystem is described in more detail in the next section.

The **video bus controller** provides access to the external memory for both the video processing subsystem and the microcontroller subsystem.

Finally, the **system management unit** manages the chip initialization, the clock generation, and the power management.

8.2 The CR16 Microcontroller Subsystem

The modular concept of the CompactRISC family of processors and peripheral modules allows the design of embedded microcontroller subsystems for various types of applications. Important real-world examples of such applications are a keyboard and power management controller for notebooks and information appliances, a DECT (Digital Enhanced Cordless Telephone) handset baseband controller, a Bluetooth baseband controller, and the digital color image processor (LmDvp), which has served as a test case in the work discussed in this chapter.

The microcontroller subsystem implemented on the LmDvp chip has been designed around its central component, the 16-bit CompactRISC processor core CR16. Besides this processor core, there are numerous peripheral modules, as depicted in Figure 8.3. Some of these peripherals are listed hereafter:

- I2C (Inter Integrated Circuit) bus master/slave
- Microwire/SPI (Serial Peripheral Interface)
- USART (Universal Synchronous/Asynchronous Receiver/Transmitter)
- general purpose I/O (GPIO) module
- timer
- interrupt controller
- input wake up module
- USB (Universal Serial Bus) controller
- serial debug interface (SDI)
- DMA (direct memory access)
- core bus controller (CBC)

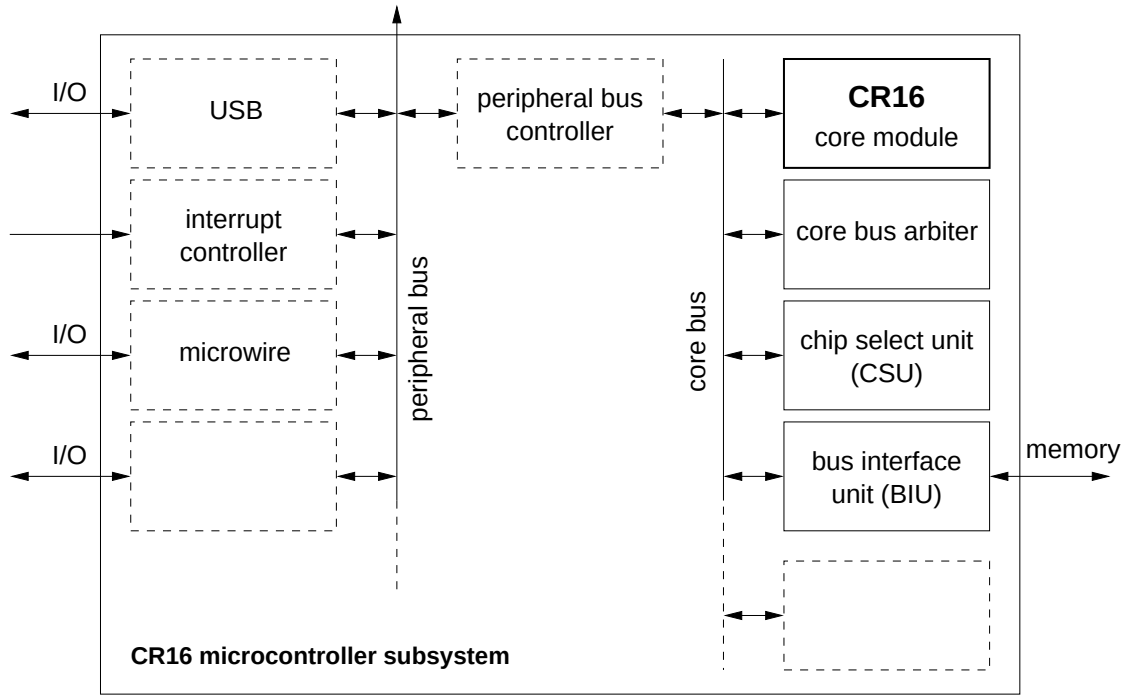

Figure 8.3: Simplified block diagram of the CR16 microcontroller subsystem.

- peripheral bus controller (PBC)
- instruction cache
- bus interface unit (BIU)
- boot ROM (read only memory)
- RAM (random access memory)

These modules communicate via two different on-chip busses. The core bus is a high-speed bus that can be used to connect performance-demanding functions to the central processing unit (CPU) such as on-chip memory, DMA channels, and additional coprocessor units. The peripheral bus is a simple, lower-speed bus for less demanding peripherals such as timers, Microwire, USB, or interrupt controller. The two busses are interconnected via the peripheral bus controller module.

In this work, the LmDvp CR16 subsystem has been simplified to consist of only its very essential components, which are the CR16 core, the core bus arbiter, the chip select unit, and the bus interface unit. In this reduced system, only the CR16 core module has been synthesized down to the gate level, while RTL Verilog models have been used for all other modules. Clearly, the CR16 processor core is the key component in the CompactRISC

microcontroller subsystem. Therefore, it was an obvious decision to focus on the core module in this work. The CR16 core is described in more detail in the next section.

8.3 The CR16 Processor Core Module

The CompactRISC family of processor cores has been specifically designed to meet the requirements of typical embedded systems. As opposed to processors that are developed to serve as CPUs in workstations or personal computers, processor cores that will be used in application specific embedded systems do not have to be optimized to achieve the highest possible performance. The key criteria that have driven the development of the CompactRISC architecture are the following:

- provide suitable performance to meet embedded application needs
- low cost
- low power consumption
- support on-chip memory with priority over external memory
- small code and data size
- low design complexity and small die size
- portability/synthesizability

The CompactRISC architecture has been developed to be scalable from 16 to 64 bit, with a common high-level language development and debug environment provided for all derivates. In this work, the 16-bit version named CR16 has served as a test case for the investigation of DSV logic synthesis issues.

8.3.1 The CR16 Architecture

Although its name implies that the concept of reduced instruction set computers (RISC) has been adopted in the development of the CompactRISC family of processor cores, the architecture really is typical of processors from the post-CISC/RISC era, where the two originally quite different concepts of CISC (complex instruction set computer) and RISC have converged. Today, practically all commercial processors combine elements of both.

The CR16 implements a simple load/store instruction set, which consists of approximately 50 instructions and supports the following five addressing modes:

REGISTER The operand is the content of a register.

IMMEDIATE The operand is a constant encoded in the instructions displacement field.

RELATIVE The operand is located in the memory; the address is the sum of the content of a register and a constant encoded in the displacement field.

FAR-RELATIVE The operand is located in the memory; the address can be determined from the contents of two registers and a constant encoded in the displacement field.

ABSOLUTE The operand is located in the memory; the address is a constant encoded in the displacement field.

Memory can be accessed only through load/store and a small number of bit manipulation instructions. All of the other instructions operate on the contents of registers or on constant values encoded in the instruction word. This makes the instruction set somewhat RISC-like. However, as opposed to pure RISC machines, the instruction length is not fixed – it can be 2, 4, or 6 bytes – and not all instructions can be executed in one clock cycle. Also, the data width is variable and can be 8, 16, or 32 bit. Finally, the CR16 provides a relatively large number of internal registers, which is again typical of RISC-like load/store architectures. Details of the register set are given in Figure 8.4.

A shallow three-stage integer pipeline is used for concurrent processing of instructions. With this pipeline, the CR16 can fetch one instruction while simultaneously decoding a second and executing a third instruction. In the first pipeline stage, an instruction is fetched from the memory. The instruction is then passed to the second stage, where it is decoded. The decoded information is subsequently used by the control logic for generating the signals required for controlling the data-path in the third stage, the instruction execution stage.

As can be seen from Figure 8.5, the data-path is composed of the register file containing the registers described before, the arithmetic logic unit (ALU), a barrel shifter, and a hardware multiplier for fast 16-bit integer multiplication. During the execution of different types of instructions, the following actions are taken. In the case of arithmetic or logic instructions, the ALU, the multiplier, or the shifter computes the results which are then written to the destination register. For load and store instructions, the ALU computes the memory address and the shifter aligns the data as necessary. For load instructions, the operand is then read from the memory and stored in the destination register, while, for store instructions, data is transfered from the selected register to the memory. In the case of branch or jump instructions, the ALU computes the target address and stores it in the program counter (PC), which is actually one of the registers in the register file.

In order to resolve any dependencies between consecutive instructions, the following policy is strictly followed during program execution. The CR16 fetches an instruction only after all previous instructions have been completely fetched. Also, data read and write operations

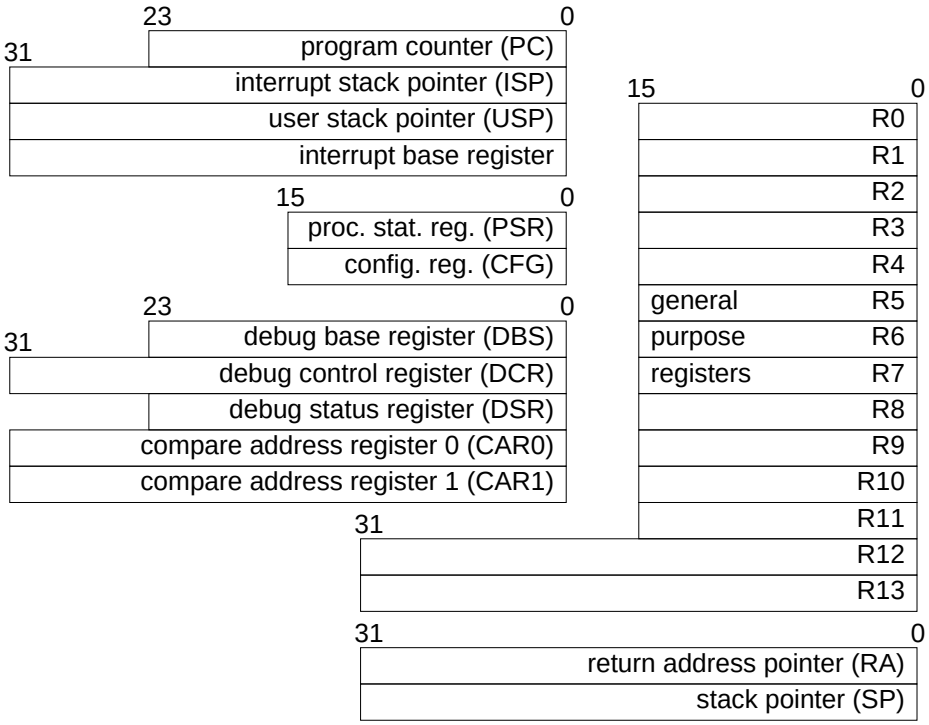

Figure 8.4: The CR16 register set.

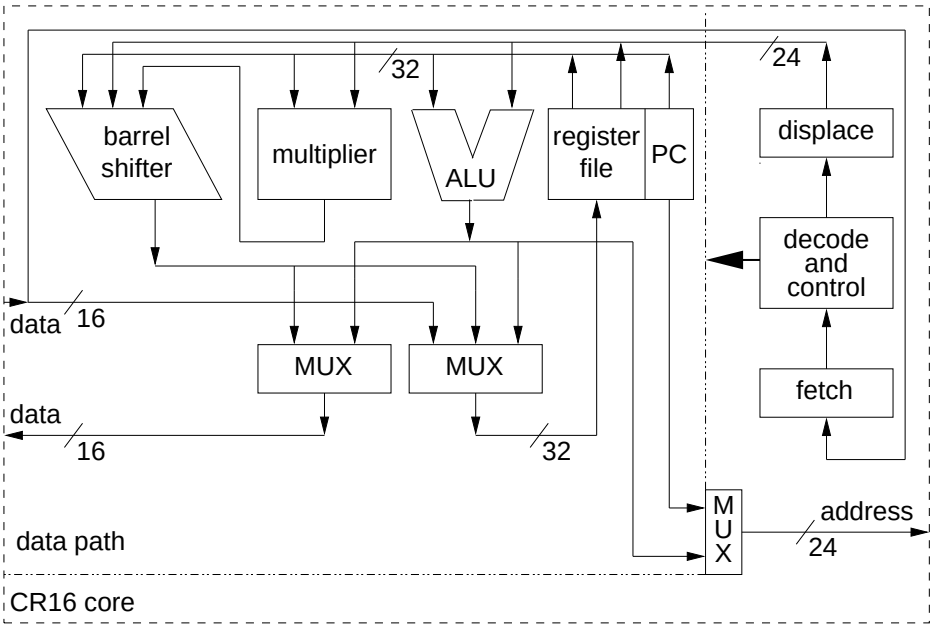

Figure 8.5: Block diagram of the CR16 core module architecture.

associated with one instruction are carried out only after all data read and write operations associated with previous instructions have been completed.

The simplicity of the instruction set, the shallow pipeline, and the absence of floating point units result in reduced design complexity which directly translates to low power, small die size, and low cost. The variable instruction length and data width assure efficient use of the usually limited amount of on-chip memory space.

The CR16 core module is available as a synthesizable RTL Verilog HDL description, which makes it portable to various technologies and adaptable to different performance requirements.

8.3.2 Clock Gating

The concept of local clock gating, which has been introduced in Section 3.4, is used in the CR16 core for deactivating functional units in the data-path when they are not needed for the execution of a certain instruction. The necessary clock enable signals are generated by the instruction decoder. The clock gating strategy has been modeled in the RTL Verilog HDL code. If necessary, it can be disabled by means of a switch in the HDL code.

8.3.3 Design for Testability

The task of determining whether chips are fully functional is highly complex and can be very time-consuming. However, when faulty chips pass an improperly designed test, they can cause system failures and enormous difficulty in system debugging resulting in tremendously increasing cost. For these reasons, design for testability has become an important issue [55].

The objective of design for testability is to maximize controllability and observability. The controllability of a circuit is a measure of the ease or difficulty with which a specific signal value can be established at each internal node by setting values at the primary input ports. The observability is a measure of the ease or difficulty with which the signal value at any internal node can be determined by observing the primary output ports. The controllability and observability can be improved by making internal nodes accessible from the primary inputs and outputs. This could be achieved by simply providing direct access to internal nodes via additional input and output ports at the expense of significantly increased packaging cost. A better concept is the scan test technique, where the registers in a sequential circuit are used as control and observation points. With the scan test technique, the testing of a sequential circuit is reduced to the problem of testing a combinational circuit.

Every sequential circuit can be partitioned into a combinational circuit and a set of registers. In circuits designed for scan test, the registers are connected to form a small number of long serial shift registers, the so-called scan paths or scan chains, using multiplexers at the input

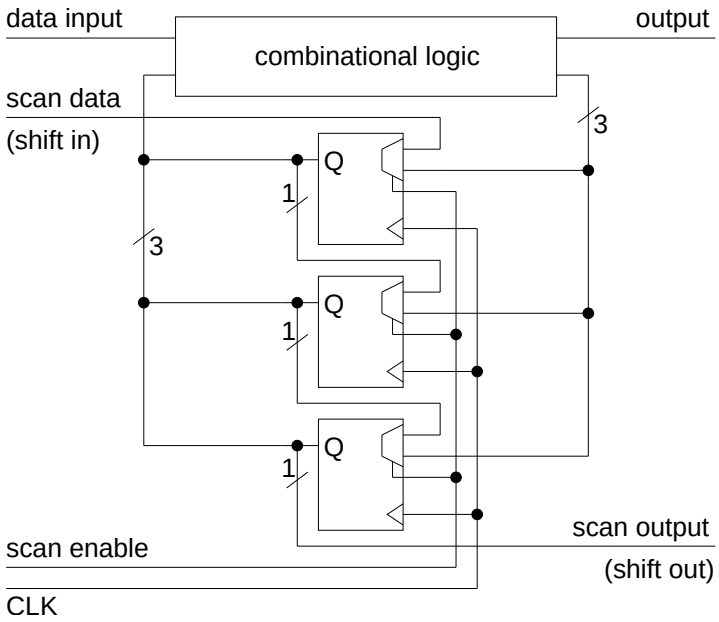

Figure 8.6: Simplified structure of a scan-testable design.

pins of the registers, a mode selection signal, and one pair of primary input and output ports per scan chain, as shown in Figure 8.6. For the sake of simplicity, the following explanations assume the existence of only one scan chain.

For scan testing, firstly, the circuit is switched to the test mode, which configures the registers as one long shift register. Secondly, a test input state vector is shifted into the scan chain. Thirdly, the circuit is switched to the normal mode for the duration of one clock cycle with appropriate data applied to the data input ports. Finally, the test mode is activated again in order to shift the resulting state vector out of and the next test input state vector into the scan chain.

Design for scan testability requires special flip-flops. Scan-flip-flops allow one of two data input pins to be selected using an input multiplexer and a scan enable input pin. For the experiments on the CR16 core, low-voltage, high-voltage and level-converting versions of scan-D-flip-flops with asynchronous preset and reset pins have been made available in the DSVL018 synthesis library, as discussed in Section 6.5. These flip-flops have one non-inverting functional output pin and one dedicated scan output pin each. This is different from the situation depicted in Figure 8.6 and is required for a proper handling of the level-conversion issue in DSV designs. Finally, in DSV designs, level converters must be inserted in the scan chains wherever needed. The modified scan test concept that has been employed in the experiments discussed in this chapter is shown in Figure 8.7.

Design for scan testability somewhat degrades the performance because of the additional delay introduced by the multiplexers at the data input pins of the scan-flip-flops. In spite of

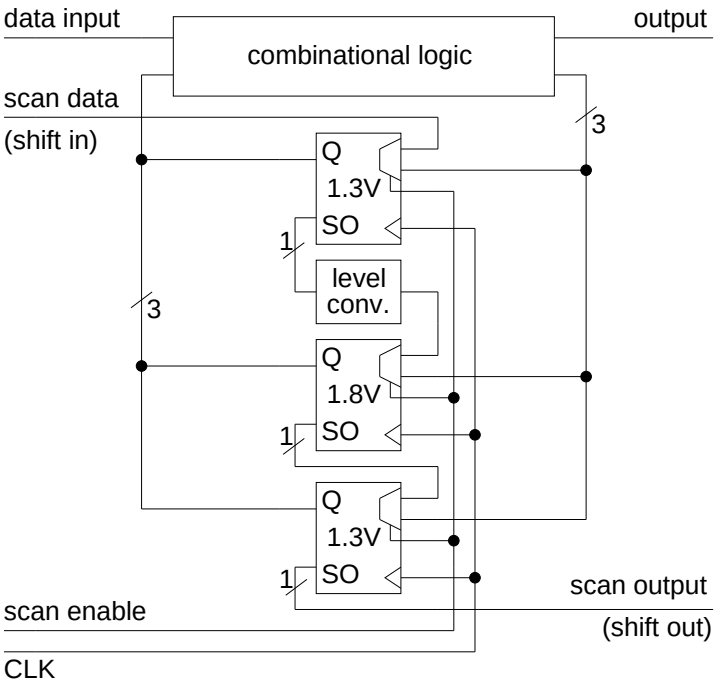

Figure 8.7: Concept of DSV design for scan-testability.

that, design for scan testability is standard for the design of CompactRISC microcontrollers and has, therefore, been part of all experiments covered in this chapter.

8.4 Technology, Library, and Operating Conditions

The CR16 processor core has been mapped to an extended version of NATIONAL SEMICONDUCTOR'S CMOSX-9 (0.18 µm CMOS technology for 1.8 V nominal supply voltage) library, which is named DSVL018 in this document. This extended library contains level-converting flip-flops and additional low voltage (1.3 V) versions of all conventional combinational and sequential cells. Two level-converting inverter and buffer cells have been added to the library as well. However, in the experiments discussed in this chapter, these cells have only been used where this is explicitly mentioned. In order to limit the library characterization effort to what is absolutely necessary, only nominal operating conditions have been considered in all experiments and investigations discussed in this chapter. A so-called custom wire-load model extracted from post-layout data on the CR16 core module realized in NATIONAL SEMICONDUCTOR'S CMOS9 technology has been added for more realistic wire load estimations. For more details of the design of level-converting flip-flops, of library modeling issues, and of the content of the DSVL018 library see Section 6.5.

8.5 Optimization Strategies and Constraints

The same basic design flow that has been used for optimizing the benchmark circuits has also been used in the experiments on the CR16 core module. Furthermore, the optimization strategies (see Figure 7.1) and constraints have been similar. Some aspects specific to the CR16 design environment, however, had to be taken into account. These aspects are discussed in the following sections.

8.5.1 Strategies and Constraints for Timing-Driven Synthesis

Before the timing-driven synthesis, the designer must decide whether to implement clock gating or not. This special feature of the CR16 design can be enabled or disabled by means of a switch in the Verilog HDL source code (START in Figure 7.1). The implementation of gated clocks should always be enabled unless there is good reason to do otherwise.

At the beginning of the actual synthesis process, the RTL design description (START) is elaborated and a so-called wire-load model is assigned to the design. The wire-load model is used for estimating interconnect capacitances and resistances in pre-layout design phases. Wire-load models are basically look-up tables that contain the wire-length as a function of the number of gate input pins driven by a net. This estimated wire-length is multiplied by a capacitance or resistance per unit length in order to obtain an estimate of the interconnect capacitance or resistance, respectively. For the experiments discussed in this chapter, a custom wire-load model has been used (see Section 8.4).

Next, the global target timing is specified. This includes the clock period and delays assigned to the primary input and output ports. The input and output delays reflect the characteristics and requirements of the system the CR16 will be embedded in. Furthermore, load capacitances are assigned to the primary output ports. Finally, all timing constraints are locally removed from so-called false paths, i.e. paths through absolutely non-timing-critical parts of the design such as software debugging support circuitry.

The CR16 core has been optimized subject to both strict and relaxed timing constraints. The strictest timing constraints have been determined in an iterative manner. First, the timing constraints have been set to values that are too strict to be met and logic synthesis has been carried out. Next, the timing constraints have been relaxed depending on the timing violations, and logic synthesis has been carried out again. These steps have been repeated until the strictest timing constraints that can be met have been found. The relaxed constraints have been set to 120% and 132% of the shortest possible critical path delay. The second value denotes the shortest possible critical path delay of an all-low-voltage implementation of the CR16, i.e. the supply voltage can be scaled all the way down to V_{DDL} if the delay is not more critical than this.

Besides timing constraints, generally, zero area constraints are specified in order to find the smallest possible implementations. In the cost function that is effective during power

optimization, timing has absolute priority over area, so that, with these constraints, area is minimized without sacrificing the performance.

After constraining the design, an appropriate subset of cells from the DSV synthesis library is selected. For the initial timing-driven synthesis (STEP 1A/B in Figure 7.1), this subset of cells contains high voltage cells while low voltage and level-converting cells are disabled. Only if clock voltage scaling is enabled, the level-converting flip-flops must be selected instead of the high voltage flip-flops. Clock voltage scaling must be taken into account already during timing-driven synthesis because of its impact on the performance.

The following logic synthesis step (STEP 1A/B in Figure 7.1) results in a timing- and area-optimized gate-level implementation of the CR16 core. The scan-flip-flops contained therein – no other flip-flops are used – are automatically arranged in a predefined number of scan chains. In this work, the number of scan chains has always been three.

8.5.2 Gate-Level Simulation and Power Analysis

Gate-level power analysis and optimization require detailed information on the switching activities of all circuit nodes. This information can be obtained from gate-level simulation.

The selection of input pattern (stimuli) for the gate-level simulation is an important task with large impact on the quality of the power optimization and analysis results. Especially in the case of a general purpose microprocessor, this is very difficult since the switching activities depend largely on the software which will eventually be executed. The more the designer knows about the application and the more regular the software and its execution characteristics are, the better the optimization results will be. In this project, the CR16 core module has been treated like any general purpose processor, since information on the software that will be executed in the digital video processor system was lacking.

A generic program comprising an intuitively chosen set of data-path and memory intensive instructions has been created as input pattern for all gate-level simulations. The idea behind this set of instructions is to generate switching events within all parts of the CR16 core module, thus, triggering power optimization for the entire design. As shown in Figure 8.8, the program contains a number of essential instructions like load, store, bit manipulation and arithmetic operations that frequently occur in real software.

In most experiments, gate-level power analysis has been carried out in the pre-layout design phase. Interconnect capacitances have been estimated on the basis of the custom wire-load model mentioned before. Post-layout power analysis has been performed in a few selected experiments, firstly, in order to evaluate the relative accuracy of the pre-layout power analysis (see Section 8.6.1) and, secondly, for the purpose of a more realistic clock tree analysis (see Section 8.6.4).

```
.macro CHECKW val,regch,regadd #param.: 1 const. , 2 regs.
  cmpw {val},{regch} #comp. const. value with content of
  bne  end_of_test   #first reg. and quit if not equal
  addw $1,{regadd}   #increment content of second reg.
.endm

storw  r1,8(r0)      #store content of reg. r1 to mem.
                     #dest. addr. = (content of r0) + 8
storw  r1,9(r0)
storm  $4            #store 4 regs. (r2-r5) to memory
                     #dest. addr. = content of r1
push   $4,r1         #save 4 regs. (r1-r4) to prog. stack
cbitw  $5,8(r0)      #clear bit 5 at (content of r0) + 8
cbitw  $15,6(r0)
sbitw  $5,8(r0)      #set bit 5 at (content of r0) + 8
sbitw  $15,6(r0)
loadw  0x001000,r2   #load from mem. at 0x001000 to r2
loadw  0x001002,r3
loadw  0x001004,r4
loadw  0x001006,r5
CHECKW $0x0123,r2,r7 #(see above)
CHECKW $0x4567,r3,r7
CHECKW $0x89AB,r4,r7
CHECKW $0xCDEE,r5,r7
mulsw  r10,(r3,r2)   #mult. signed r10 by r2 and store
                     #result in r3 and r2
end_of_test
```

Figure 8.8: Assembly language program used as stimulus for gate-level simulation.

8.5.3 Strategies and Constraints for Power Optimization

The strategies that have been used for optimizing the power consumption of the CR16 core module can also be found in Figure 7.1. Single and dual supply voltage power optimization (STEP 2A and STEP 2B respectively) have been performed after the initial timing-driven logic synthesis. The timing constraints have been the same as those that have already been effective in timing-driven synthesis. Note that DSV power optimization (STEP 2B) includes simultaneous SSV power optimization.

In the case of relaxed delay constraints, GSVS has been carried out as an alternative to DSV power optimization. The procedure that has been used here is basically the same as the one described in Section 7.4. The RTL design has been synthesized subject to strict

timing constraints (STEP 1B), followed by conventional SSV power optimization (STEP 2A) and GSVS (STEP 2C).

The dynamic power constraints have always been set to zero. In the cost function, which is effective during power optimization, timing has absolute priority over power and power has absolute priority over area, so that, with these constraints, power is optimized as far as possible without degrading the performance and area is minimized without sacrificing power and performance.

8.6 Results

8.6.1 Analysis of a Typical CR16 Implementation

The CR16 core module has been synthesized to NATIONAL SEMICONDUCTOR'S 0.18 µm technology (CMOS9) for operation at a nominal supply voltage V_{DD} of 1.8 V (see Section 8.4). Performance has been given highest priority and, thus, the primary objective has been to minimize the clock period. This has been accomplished by means of an iterative synthesis strategy as discussed in Section 8.5.1. Following the common standards for CR16 implementations, the module has been prepared for scan test (see Section 8.3.3) and gated clocks have been enabled for dynamic power reduction (see Section 8.3.2).

Timing and power characteristics. The results of the timing-driven synthesis subject to the strictest timing constraints (STEP 1A/B in Figure 7.1) both before and after place and route are summarized in Table 8.1. Obviously, the power consumption of the core module has been underestimated in pre-layout gate-level power analysis. This is due to the shortcomings of wire-load-model-based interconnect capacitance estimation. It is also worth noting that only two additional driver cells have been inserted into the clock network during clock tree synthesis in the layout design phase. This means that most timing requirements have been met by proper sizing of the clock gating elements that already existed in the clock network in the pre-layout design phase.

The distribution of the total dynamic power consumption between different sections of the CR16 core module is shown in Figure 8.9. The data has been determined by pre-layout gate-level power analysis. At the top level, the data path (dp) can be identified as the main power consumer. It accounts for more than 60% of the total dynamic power consumption while the bus state machine (bsm), the execution state machine (esm), and the instruction fetch, decode and displacement unit (queue, qu) together account for only 35%. The data-path can be further divided into the program counter (pc), some debug circuitry (dbg) and the data-path core (core). The latter mainly consists of the register file (rf) and the arithmetic units, i.e. the ALU (alu), the barrel shifter (bsh), and the multiplier (mul). The data-path core consumes almost 90% of the total dynamic power in the data-path. Finally,

	Pre-layout analysis	Post-layout analysis
Voltages and timing:		
supply voltage	V_{DD}	
clock voltage	V_{DD}	
clock period	9.9 ns	
Dynamic power (not optimized):		
P_{dyn}	2.81 mW	4.40 mW
P_{clk}	0.16 mW	0.31 mW
P_{clk}/P_{dyn}	6%	7%
Number of cells:		
total (comb./FF)	14038 / 1319	
clock driver/gating cells	115	117

Table 8.1: Performance, power consumption and complexity of a typical SSV implementation of the CR16 processor core.

a very important result of this analysis is that only 6% of the total dynamic power of the processor core can be attributed to the clock network, which includes the driver and gating elements and the actual clock nets. This small number is the consequence of a highly efficient clock gating strategy, as another analysis presented in Section 8.6.4 shows. Figure 8.10 shows the results of a power distribution analysis carried out after place and route. The numbers are not significantly different from the pre-layout results discussed before. This holds even for the clock network because of the negligible number of cells inserted during the clock tree synthesis.

The comparison of the pre- and post-layout results in Table 8.1 supports the widespread opinion that pre-layout power analysis at the logic level is inaccurate in terms of absolute values. Nevertheless, this type of power analysis can be considered reasonably accurate in terms of relative values. This is another accepted opinion, which is supported by the comparison of the pre- and post-layout results from Figures 8.9 and 8.10.

Optimization potential. The amount of power reduction that can potentially be achieved by gate sizing, DSVS, and other techniques that trade off delay against power depends on the timing slack that exists in the timing-optimized implementation of the design. Figure 8.11a shows the results of a type-1 slack analysis (see Section 7.5.3 for details on the procedure) for the timing-optimized implementation of the processor core in the form of a histogram with the percentage of cells on the vertical axis and the slack normalized to the clock period on the horizontal axis. The horizontal axis is divided into seven intervals.

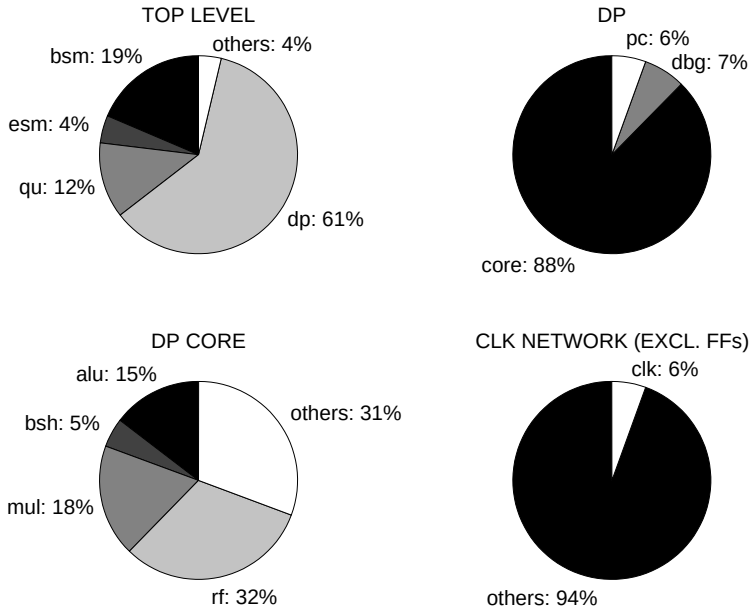

Figure 8.9: Power distribution within the CR16 processor core (pre-layout).

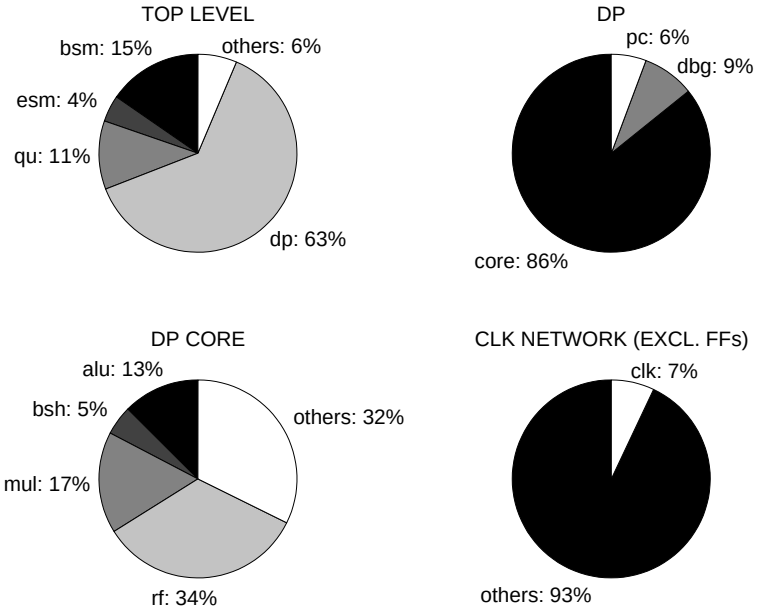

Figure 8.10: Power distribution within the CR16 processor core (post-layout). A comparison with Figure 8.9 confirms that the pre-layout power analysis is reasonably accurate in terms of relative values.

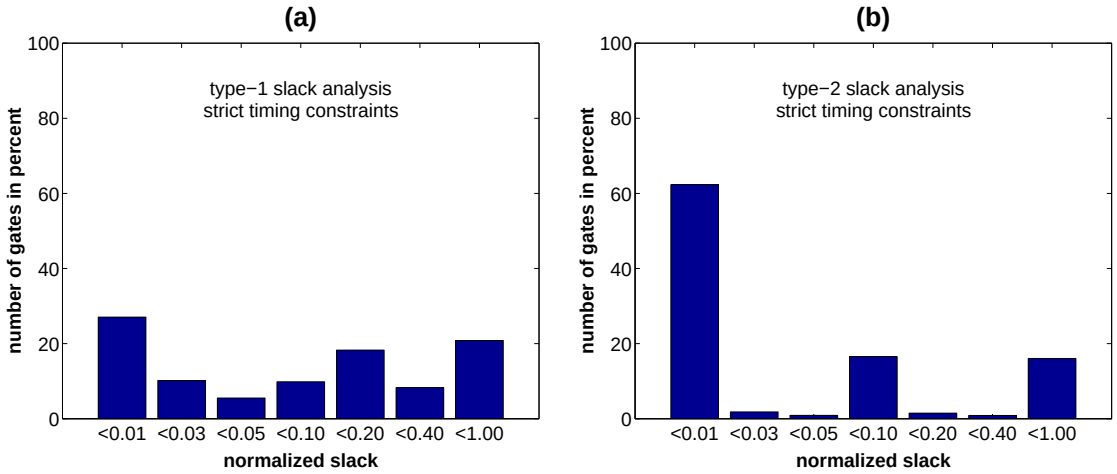

Figure 8.11: Slack statistics for the CR16 core (timing-optimized SSV implementation): (a) Straight-forward slack analysis according to Chen et al.; (b) modified analysis taking into account the restrictions imposed by the level-conversion issue.

The width of these intervals increases from smaller to larger slack. The height of the bars is proportional to the number of gates that have been assigned a slack value from the respective slack interval. Obviously, there is a large number of critical gates in the design. However, the number of non-critical gates is still significant and, hence, there is a potential for noticeable power reduction through gate sizing.

Regarding DSV logic synthesis these statistics do not adequately describe the optimization potential, because the level-conversion issue is ignored. This is taken into account in the type-2 slack analysis procedure introduced in Section 7.5.3, where every gate in the netlist is assigned the slack of the most critical gate in its fan-out. Figure 8.11b shows the results of the type-2 slack distribution analysis for the timing-optimized implementation of the processor core. Compared with the the type-1 analysis, a significantly smaller number of cells has been identified as being non-critical. Thus, the potential for power reduction through DSVS must be expected to be relatively small.

8.6.2 Power Optimization Subject to the Strictest Timing Constraints

Optimization without clock voltage scaling. Starting from the typical implementation of the CR16 core module described in the previous section, power optimization subject to the strictest timing constraints has been carried out, firstly, using the conventional SSV optimization techniques (STEP 2B in Figure 7.1) and, secondly, using SSV and DSV optimization techniques (STEP 2A or 2C in Figure 7.1). Here, a high voltage clock signal has been used. The results are summarized in Table 8.2, where each pair of parenthesized numbers is a cross-reference between the power reduction in percent (achieved in a certain

	Before power opt.	After SSV power opt.	After DSV power opt.
Voltages and timing:			
supply voltage(s)	V_{DD}	V_{DD}	V_{DD} / V_{DDL}
clock voltage	V_{DD}		
clock period	9.9 ns		
Total dynamic power:			
$P_{dyn}^{(\text{ref.})}$	2.81 mW$^{(1)}$	2.49 mW$^{(2)}$	2.39 mW$^{(3)}$
$\Delta P_{dyn}/P_{dyn}^{(\text{ref.})}$	—	-11%$^{(1)}$	-4%$^{(2)}$ / -15%$^{(1)}$
Number of cells:			
total (comb./FF)	14038 / 1319	13157 / 1319	12995 / 1319
minimum size	66%	77%	76%
low voltage (comb./FF)	—	—	$<$ 1% / 0
level-converting FF	—	—	20%

Table 8.2: Results of power optimization subject to the strictest timing constraints. Numbers in parentheses relate a power reduction in percent to a reference value given in milliwatts. Example 1: $(2.49-2.81)/2.81 = -11\,\%$. Example 2: $(2.39-2.81)/2.81 = -15\,\%$.

power optimization step) and the power consumption in milliwatts before the respective power optimization step. Again, note that the absolute power values given in the table are the results of pre-layout gate-level power analysis and, hence, can only be rough estimates of the actual power consumption of the processor.

The SSV optimization has reduced the total dynamic power consumption by 11% compared with the situation before power optimization. This power reduction can be attributed mainly to gate sizing and logic restructuring, as indicated by the increasing number of minimum sized cells and the decreasing total number of cells.

As expected from the slack analysis presented in the previous section, the additional power reduction of 4% (compared with the results of SSV power optimization) that has been achieved using DSV logic synthesis is small. Although 20% of all flip-flop cells have been replaced with their level-converting counterparts, this has enabled voltage scaling for less than 1% of all cells. Also, part of the power reduction is probably due to logic restructuring, as indicated by the slight reduction in the total number of combinational cells, which can also take place during DSV power optimization.

Low voltage flip-flops have not been used. Thus, no additional level converters that cause power and area overheads are needed in the scan chains.

The overall power reduction due to simultaneous SSV and DSV power optimization has been 15% (see column four of Table 8.2).

Optimization including clock voltage scaling. In a second set of experiments, the voltage scaling approach has been extended to the clock network in order to achieve additional power reduction. For this purpose, the use of high voltage flip-flops has been disabled from the beginning, so that the SSV implementations of the CR16 core module before and after SSV power optimization contain only level-converting flip-flops and the DSV implementation contains only level-converting and some low voltage flip-flops, as shown in the last two rows of Table 8.3. Under these circumstances, the signal level in the clock network can be safely reduced from V_{DD} to V_{DDL}.

Since level-converting flip-flop cells are often slower than conventional flip-flops, the substitution of high voltage flip-flops with their level-converting counterparts in circuits that are subject to the strictest timing constraints usually degrades the overall performance. In the case of the CR16 core module, the performance penalty has been small; the clock period has increased by only 2% from 9.9 ns to 10.1 ns (see Tables 8.2 and 8.3). There has, however, been a power overhead of 5%, as shown in the second column of Table 8.3. Even after reducing the signal level in the clock network, there has still been a power overhead of 2% remaining, as indicated in the third column. This overhead is neither due to a generally larger power consumption of the level-converting flip-flops, as indicated by the data that has been presented in Section 6.5.2, nor due to massive logic parallelization, as observed in the case of several benchmark circuits. The number of flip-flops with larger driving strength has also not increased. The overhead is simply due to the flip-flops being operated in a different circuit environment which changes, for instance, the output loads and the input signal transition times. Note that, in Table 8.3, numbers in parentheses may be parts of cross-references between Table 8.2 and Table 8.3.

The fourth column of Table 8.3 contains the results of SSV power optimization. The power consumption has been reduced by 11% compared with the same design before power optimization. This is again mainly due to gate sizing and logic restructuring, as indicated by the increasing number of minimum size cells and the decreasing total number of cells. Compared with the power optimized SSV design that uses a high voltage clock signal (see Table 8.2), there has still been a power overhead of 2%.

According to column five of Table 8.3, the DSV power optimization has resulted in 7% lower power compared with the same design after SSV power optimization. Obviously, the efficiency of the DSVS technique has been improved by the large number of level-converting flip-flop cells. However, the reduced clock signal level and the higher efficiency of the DSVS technique have just compensated for the power overhead; the power consumption of the DSV implementation with low voltage clock signal (see Table 8.3) is only 1% lower than that of the DSV implementation with high voltage clock signal (see Table 8.2). The total area occupied by the flip-flop cells in the DSV implementation with low voltage

	Before power opt.		After SSV power opt.	After DSV power opt.
Voltages and timing:				
supply voltage(s)	V_{DD}			V_{DD} / V_{DDL}
clock voltage	V_{DD}	V_{DDL}		
clock period	10.1 ns			
Total dynamic power:				
$P_{dyn}^{(\text{ref.})}$	2.94 mW[4]	2.88 mW[5]	2.55 mW[6]	2.37 mW
$\Delta P_{dyn}/P_{dyn}^{(\text{ref.})}$	—	-2%[4]	-11%[5]	-7%[6]
Comparison with implementation without clock voltage scaling:				
$\Delta P_{dyn}/P_{dyn}^{(\text{ref.})}$	+5%[1]	+2%[1]	+2%[2]	-5%[2] / -1%[3]
Number of cells:				
total (comb./FF)	13835 / 1319		12927 / 1319	12744 / 1319
minimum size	56%		68%	69%
low voltage (comb./FF)	—		—	11% / 1%
level-converting FF	100%		100%	99%

Table 8.3: Impact of clock voltage scaling on the results of DSV power optimization subject to the strictest timing constraints. For references (1), (2) and (3) see Table 8.2. Example 1: $(2.94-2.81)/2.81 = +5\,\%$. Example 2: $(2.37-2.55)/2.55 = -7\,\%$.

clock – area values are not shown in the table – is 19% larger compared with the DSV implementation with high voltage clock, which agrees with the area overhead of individual cells (see Section 6.5.2). This results in 9% larger cell area for the entire circuit. Additional level-converters in the scan chains that would further increase the total cell area are not required if only level-converting and low voltage flip-flops are used.

Clearly, clock voltage scaling has not led to a significant improvement. The additional power reduction achieved through DSV power optimization compared with the results of SSV optimization has been increased only slightly from 4% to 5% (see rightmost columns of Tables 8.2 and 8.3). The reason is that the clock network accounts for only 7% of the total dynamic power (see Table 8.1) and, hence, even a significant reduction of the power in the clock network (e.g. 50%) results in very little reduction of the total power (e.g. 3%).

Impact of power optimization on the slack distribution. Figure 8.12a shows how the slack distribution in the CR16 core module has been changed by SSV and DSV power optimization without clock voltage scaling. There are three bars associated with each slack

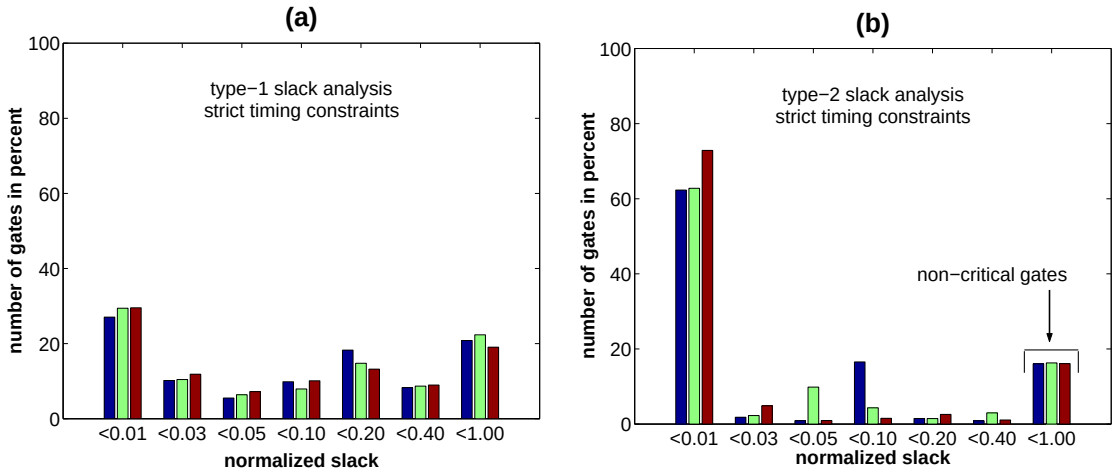

Figure 8.12: Slack statistics for the CR16 before and after power optimization. Three bars are associated with each slack interval, one corresponding to the results of timing-driven synthesis (left), one representing the results of SSV power optimization (middle), and one describing the situation after DSV power optimization (right).

interval. In each group of three bars, the left bar corresponds to the situation after timing-driven synthesis, the middle bar represents the results of SSV power optimization, and the right bar describes the situation after DSV power optimization. The graph shows a slight increase in the number of critical gates due to SSV power optimization. The effect of DSV power optimization on the number of most critical gates, i.e. combinational cells with a normalized slack of less than 0.01, is hardly noticable in this graph, but it becomes obvious in the results of the modified slack analysis where the level conversion issue is taken into account. In Figure 8.12b it is clearly visible that the number of most critical gates has increased in the DSV power optimization step.

In all experiments covered in this chapter, roughly 15% to 20% of all gates belonged to the class of gates with the largest slack, i.e. a normalized slack between 0.4 and 1.0, even after SSV and DSV power optimization. This can be seen in Figures 8.12b, 8.13b, and 8.13d. In the remainder of this chapter, these gates are simply referred to as non-critical gates. More than 90% of these non-critical gates are located in the execution state machine (esm) and the decode and displacement unit (qu). Spot checking on the non-critical gates has revealed several reasons for positive slack remaining after SSV and after DSV power optimization.

Possible reasons for positive slack of a cell not being exploited by gate sizing:

- The cell already has the smallest available size.
- The cell's size is the same as that of a cell with minimal driving strength and, hence, the area constraint does not force the tool to reduce the driving strength, AND ...

- ...the output pin has zero switching activity, which means that power optimization is not triggered for the cell (stimuli show insufficient node coverage).
- Maximum load capacitance or maximum output transition time constraints prohibit the use of a smaller cell.

Possible reasons for positive slack of a cell not being exploited by DSVS:

- The cell is operated at low voltage already.
- The cell is ”blocked” by a high-voltage flip-flop in its fan-out (level-conv. issue).
- The output pin has zero switching activity and, hence, power optimization is not triggered for the cell (stimuli show insufficient node coverage).
- Maximum load capacitance or maximum output transition time constraints prohibit the use of a lower supply voltage.
- The cell is ”blocked” by another non-critical cell in its fan-out when one of the two preceding conditions is true for the ”blocking” cell (level-conversion issue).

The optimization potential that has not been exploited because of the reasons stated above is negligible, as the following analysis of the properties of non-critical gates shows.

Immediately after timing-driven synthesis subject to the strictest timing constraints without clock voltage scaling, 2224 non-critical gates (16% of all gates) have been counted and 1767 (79%) of these 2224 gates have been of minimum size. Another 452 (20%) have been of larger driving strength but minimum cell size. Only five gates have been of larger than minimal size, which is a negligible number.

After DSV power optimization, there have still been 2067 non-critical gates (16% of all gates) and 1825 (88%) of these 2067 gates have been of minimum size. Another 233 (11%) gates have been of larger driving strength but minimum cell size. This is because of insufficient node coverage. Power optimization is not triggered for 211 out of these 233 gates because of zero switching activity at the output pin. Only 19 (1%) out of the 2067 gates have been operated at V_{DDL}. The reason for this is that 2045 (99%) non-critical gates are ”blocked” by at least one high-voltage flip-flop in the fan-out paths.

The dynamic power consumption caused by all non-critical gates together has been reduced from 0.12 mW before power optimization to 0.10 mW after DSV power optimization. This improvement is negligible in comparison with the total power consumption of the CR16 core module.

With clock voltage scaling, level-converting flip-flops are forced into the design instead of high voltage flip-flops. The non-critical gates cannot be ”blocked” by high-voltage flip-flops anymore, which enables voltage scaling for a larger number of gates.

After DSV power optimization, 1874 non-critical gates (15% of all gates) have been identified. Supply voltage scaling has been applied to 1031 (55%) of these 1874 gates. The main reason for the remaining 843 non-critical gates not being operated at low voltage is that of insufficient node coverage. Power optimization is not triggered for 667 (79%) of the non-critical gates because of zero switching activity at the output pin.

The power consumption caused by all non-critical gates together has been reduced from 0.11 mW to 0.05 mW. This is a significant improvement over the case without clock voltage scaling. However, the power reduction is still negligible in comparison with the total dynamic power consumption of the CR16 processor core.

Level converting cells in combinational logic paths. In contrast to most benchmark circuits, the CR16 exhibits relatively large combinational logic path delays (9.9 ns). Since the extra delay introduced by inserting level converters into long paths is relatively small (typically 0.35 ns), a number of experiments have been carried out in order to find out whether using level-converting cells along combinational logic paths is feasible in the case of the CR16 processor core.

An inverting level-converter cell (INVLC) and a non-inverting buffer type level-converter cell (BUFLC) have been included in the DSVL018 synthesis library (see Section 6.5.2). With this extended library, power optimization without clock voltage scaling subject to the strictest timing constraints has been repeated.

It has been observed that only one INVLC cell and four BUFLC cells have been used and the final power consumption has not been reduced any further. Obviously, the insertion of level-converting cells into combinational logic paths is not efficient in this methodology. The most likely reason is that the underlying algorithms have, of course, not been developed for inserting extra cells for the sole purpose of level conversion.

8.6.3 Power Optimization Subject to Relaxed Timing Constraints

The CR16 core module has also been synthesized and optimized subject to relaxed timing constraints. Under these circumstances, larger slack and, hence, a larger optimization potential have been expected. The results of these experiments are summarized in columns three to five of Table 8.4. In column two of this table, the results of the optimization subject to the strictest timing constraints (see Table 8.2) are given once again.

Single and dual supply voltage power optimization. A first set of experiments has been carried out allowing for a 20% longer clock period, i.e. 11.9 ns instead of 9.9 ns (see column three). In this case, power reductions of 13% and 3% have been achieved by SSV and DSV power optimization, respectively. The overall power reduction due to simultaneous SSV and DSV power optimization has been 16%. In another experiment, the clock period has

	Strict timing	Relaxed timing		
Voltages and timing:				
supply voltage(s)	V_{DD} / V_{DDL}			V_{DDL}
clock voltage	V_{DD}			V_{DDL}
clock period	9.9 ns	11.9 ns	13.1 ns	
Before power optimization:				
$P_{dyn}^{(\text{ref.})}$	2.81 mW[1]	2.73 mW[7]	2.72 mW[9]	1.35 mW[12]
no. of comb. cells	13978	13206	10519	13951
min. size cells	66%	70%	60%	88%
After SSV power optimization:				
$P_{dyn}^{(\text{ref.})}$	2.49 mW[2]	2.38 mW[8]	2.40 mW[10]	1.18 mW
$\Delta P_{dyn}/P_{dyn}^{(\text{ref.})}$	-11%[1]	-13%[7]	-12%[9]	-13%[12] / -50%[11]
no. of comb. cells	13157	12250	10047	12775
min. size cells	77%	80%	76%	92%
After DSV power optimization:				
$P_{dyn}^{(\text{ref.})}$	2.39 mW[3]	2.30 mW	2.35 mW[11]	—
$\Delta P_{dyn}/P_{dyn}^{(\text{ref.})}$	-4%[2]	-3%[8]	-2%[10]	—
$\Delta P_{dyn}/P_{dyn}^{(\text{ref.})}$ (tot.)	-15%[1]	-16%[7]	-14%[9]	—
no. of comb. cells	12995	12131	10028	—
min. size cells	76%	79%	73%	—
low volt. comb. cells	$< 1\%$	1%	2%	—
low voltage FF	0	0	$< 1\%$	—
level-converting FF	20%	22%	23%	—
Total number of FF:	1319			

Table 8.4: Results of power optimization subject to varying timing constraints.

been stretched further to 13.1 ns (see column four). In this case, SSV and DSV optimization have reduced the power consumption by 12% and 2%, respectively, and the overall power reduction has been 14%.

The characteristics of this design are different from those of most benchmark circuits. Although the number of non-critical gates has increased slightly as the timing constraints have been relaxed (see Figures 8.12a, 8.13a, and 8.13c), neither the effectiveness of SSV power optimization on the one hand and DSVS on the other hand, nor the overall effectiveness of simultaneous SSV power optimization and DSVS has changed significantly.

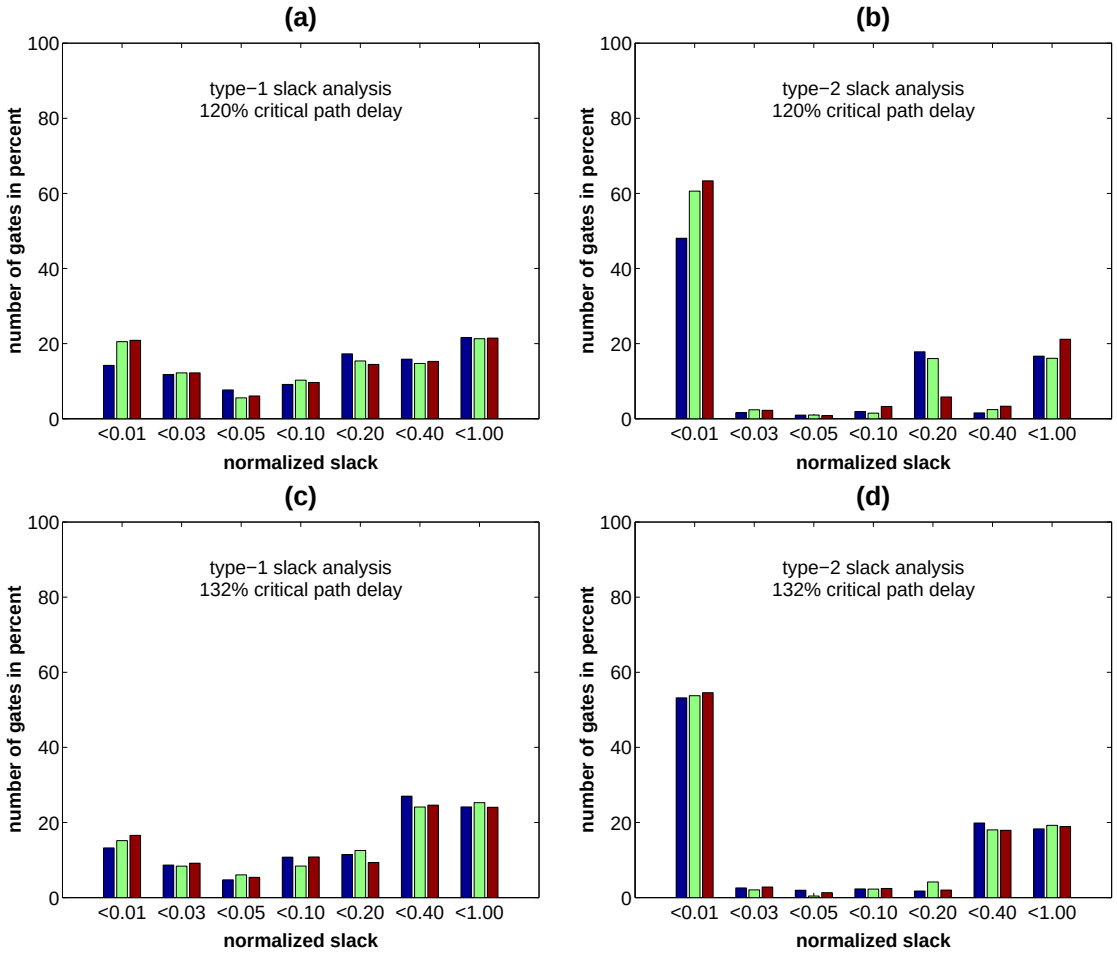

Figure 8.13: Slack statistics for the CR16 before and after power optimization. Three bars are associated with each slack interval, one corresponding to the results of timing-driven synthesis (left), one representing the results of SSV power optimization (middle), and one describing the situation after DSV power optimization (right).

Global supply voltage scaling. In a third set of experiments, global supply voltage scaling (GSVS) has been used instead of the DSV synthesis methodology. For a clock period of 13.1 ns, the supply voltage has been reduced to V_{DDL} globally without violating the timing constraints. Since the initial timing-driven synthesis and the subsequent SSV power optimization have been carried out under the strictest timing-constraints, the strategy used here is equivalent to the GSVS (I) strategy explained in Section 7.5.5. As can be seen in the rightmost column of Table 8.4, the dynamic power consumption of the design after GSVS has been about 50% lower than that of the DSV implementations discussed before. Thus, GSVS is clearly preferable in this case.

Level converting cells in combinational logic paths. The experiments regarding the feasibility of the insertion of level converters into combinational logic paths, as described in the previous section, have been repeated with relaxed timing constraints, i.e. 120% critical path delay compared with the shortest possible critical path delay.

The results again show that in this methodology the insertion of level-converting cells into combinational logic paths is not efficient. Only five INVLC cells and two BUFLC cells have been used, the increase in the number of low voltage cells has been negligible, and the final power consumption has not been reduced any further.

8.6.4 Impact of Clock Gating on DSV Logic Synthesis

The results presented in Section 8.6.2 show that, for the CR16 processor core, clock voltage scaling does not significantly improve the results of DSV logic synthesis. The primary reason is that only a very small portion of the total dynamic power can be attributed to the clock network.

For comparison, the CR16 implementation described in Section 8.6.1 has been repeated with gated clocks disabled. The results are presented in Table 8.5. In contrast to the implementation with gated clocks, the number of driver cells in the clock network has almost doubled during clock tree synthesis. This leads to a great difference between the results of pre- and post-layout analysis of the power consumption in the clock network. Pre-layout power analysis has predicted that the clock network contributes 15% to the total power consumption, while the actual contribution has been 25% after place and route, which is significantly more than in the case where clock gating has been implemented.

The effectiveness of the clock gating strategy becomes even more obvious from Table 8.6. The dynamic power in the clock network excluding the flip-flops has been reduced by 78%. Regarding all other parts of the core module, gated clocks reduce the dynamic power by 3%. This adds up to a 22% reduction of the total dynamic power consumption.

Power distribution diagrams for the CR16 core module without clock gating are shown in Figure 8.14 (before place and route) and Figure 8.15 (after place and route). The most obvious and important differences to the implementation with gated clocks are again the significantly larger power consumption in the clock network and also the larger contribution of the register file to the total power consumption.

The results presented so far indicate that clock voltage scaling is not feasible if gated clocks are used, but reasonable savings can be expected for designs where gated clocks cannot be implemented. In this work, no further experiments regarding the effect of clock voltage scaling applied to the CR16 core without gated clocks have been carried out, because, firstly, the CR16 is hardly ever implemented without gated clocks and, secondly, the pre-layout power analysis in the clock network yields invalid results (see Figures 8.14 and 8.15 as well as Tables 8.5 and 8.6). The implementation of a DSV layout flow, however, is beyond the scope of this work.

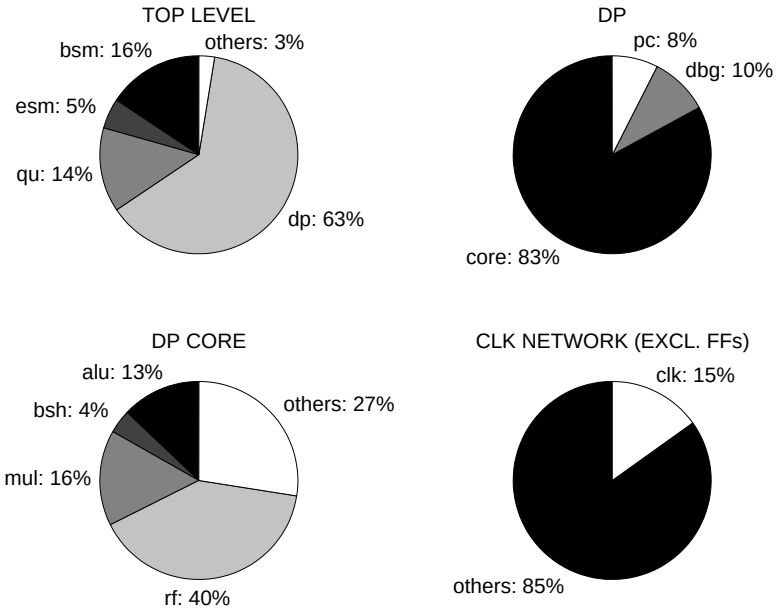

Figure 8.14: Power distribution in the CR16 without gated clocks (pre-layout).

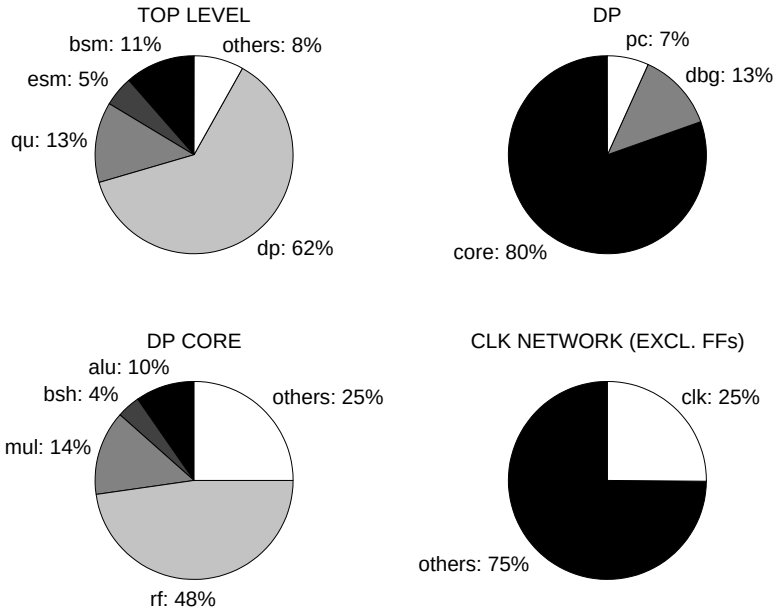

Figure 8.15: Power distribution in the CR16 without gated clocks (post-layout). Note the larger clock power value compared with pre-layout results. Major differences to Figure 8.10 are larger power consumptions in the clock network and the register file.

	Pre-layout analysis	Post-layout analysis
Voltages and timing:		
supply voltage	V_{DD}	
clock voltage	V_{DD}	
clock period	9.9 ns	
Dynamic power (not optimized):		
P_{dyn}	3.53 mW	5.65 mW
P_{clk}	0.53 mW	1.42 mW
P_{clk}/P_{dyn}	15%	25%
Number of cells:		
total (comb./FF)	15110 / 1313	
clock driver/gating cells	27	52

Table 8.5: Performance, power consumption and complexity of a CR16 SSV implementation without gated clocks.

	Clock gating		Power reduction
	disabled	enabled	
Total dyn. power P_{dyn}	5.65 mW	4.40 mW	22%
Clock net power P_{clk}	1.42 mW	0.31 mW	78%
P_{clk}/P_{dyn}	25%	7%	—
Power in other parts	4.23 mW	4.09 mW	3%

Table 8.6: Impact of clock gating on the total dynamic power, the power in the clock network and the power in other parts of the CR16 processor core (post-layout results).

8.7 Impact of DC-DC Conversion on DSV System Design

In portable electronic systems, the output voltage of the battery source is usually converted to the supply voltage required for a particular chip or a group of chips by means of a DC-DC converter. Such a converter circuit transforms a DC input voltage to a well defined DC output voltage. Most DC-DC converters contain feedback loops for voltage regulation, so as to provide a stable output voltage over a range of input voltages. This assures that the circuits supplied by the converters are always operated at the optimal voltages, regardless of the instantaneous output voltage of the battery that decreases as the battery is discharged. Apart from portable electronics, non-battery-powered systems might also contain DC-DC converters supplying chips that require other voltages than those provided by the primary

power supply unit. This includes those cases where the supply voltage of certain chips is to be (speed-)adaptively regulated (see Chapter 4). This section provides a short analysis of the efficiency of DC-DC conversion that can be achieved in the case of low power applications such as the CR16-based color image processor discussed before. Furthermore, the question of whether generating a second supply voltage for DSV design introduces power overheads will be answered.

The most popular class of DC-DC converters are the switching regulators. Figure 8.16 shows a buck converter, which is probably the most widely used switching regulator configuration [68]. The distinct characteristic of a buck converter is that the output voltage V_{out} is lower than the input voltage V_{in}. Its principle of operation is as follows. The input voltage is converted to a rectangular signal $v_{rec}(t)$ using a controlled switch, which is composed of the p-channel and n-channel transistors PSW and NSW, respectively. The low-pass filter, which is composed of a capacitor C_F and an inductor L_F, passes the DC component of $v_{rec}(t)$ to the output of the converter. The transistor PSW turns on periodically after fixed intervals T_{sw}, i.e. the switching frequency is fixed at $f_{sw} = 1/T_{sw}$. While PSW is on, it delivers a current through the inductor to the filter capacitor and to the output load. This inductor current $i_L(t)$ increases approximately linearly with time. When PSW is switched off, the n-channel transistor NSW is turned on and takes over the inductor current. In this phase, $i_L(t)$ decreases linearly so that its waveform becomes triangular in shape. The load current I_{load} is approximately constant and equal to the average inductor current. It is determined by the output voltage V_{out} and the load. When $i_L(t)$ is larger than I_{load}, the excess inductor current charges the filter capacitor. In the other case, when $i_L(t)$ is smaller than I_{load}, the capacitor is discharged, thus delivering to the load the difference between the load current and the inductor current. The ratio of the on-time of PSW to the switching period, the so-called duty cycle D, determines the average value, i.e. the DC component, of the rectangular signal $v_{rec}(t)$ and, hence, the output voltage. In the pulse width modulation (PWM) control circuitry in the feedback loop, the actual output voltage is compared with the desired value and the duty cycle is adjusted accordingly. This way, the output voltage can be adjusted to any arbitrary value V_{out} with $0 \leq V_{out} \leq V_{in}$.

Ideally, the efficiency of switching regulators is 100% [101]. Since the components used to build them are not ideal in practice, however, there are various sources of power dissipation, the most relevant of which are the following. First of all, there are the parasitic resistances of the filter components which cause so-called conduction losses. There is also a parasitic capacitance C_L at the input side of the inductor. This causes some capacitive switching losses. Another major source of conduction losses are the on-resistances of the power switch transistors. Also, since these transistors are typically large, a significant amount of capacitive power is dissipated in the gate drivers. Finally, like in any CMOS push-pull circuit structure, short-circuit currents occur in the power switches and the gate drivers.

The power consumption of the PWM control circuitry can usually be neglected in the design of high output current/power DC-DC converters [68]. Any effort of improving the efficiency is then concentrated on the so-called power train, i.e. the filter components, the

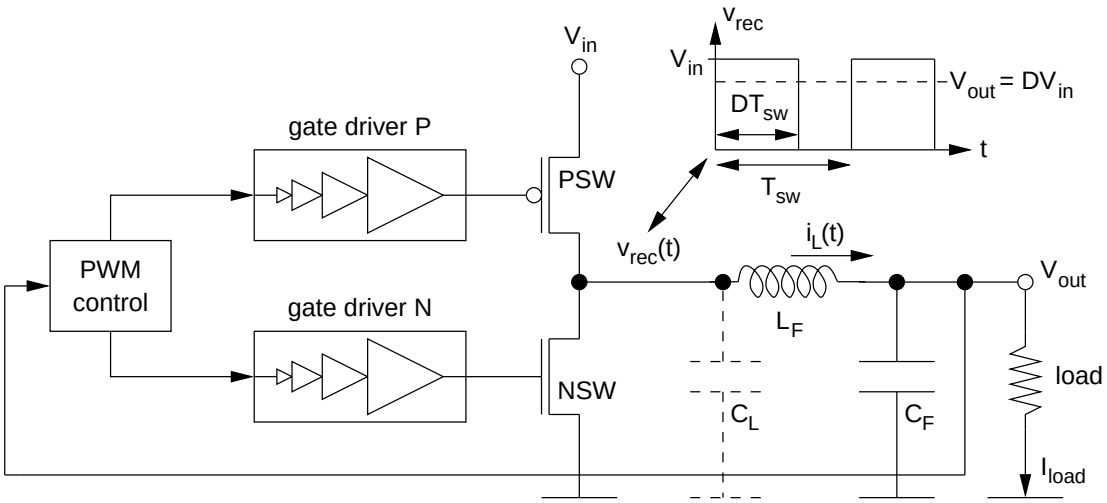

Figure 8.16: Buck-type DC-DC converter.

power switches, and the gate drivers.

The total filter losses can be reduced if the switching frequency is increased [68]. However, increasing the frequency means increasing the losses associated with the power switches and the gate drivers, so that the overall efficiency of the converter degrades with increasing switching frequency [101]. Since the filter volume increases with decreasing switching frequency, volume and cost have to be traded off against efficiency.

The on-resistance of the power switches can be reduced by increasing the width of the transistors. This comes at the cost of larger gate area and, thus, larger capacitive switching power. For a given nominal load current I_{load}, there is an optimum for the widths of the power switch transistors where the sum of the capacitive losses and the conduction losses is minimal [101]. This, in turn, means that the efficiency of a particular converter might be low if the actual load does not match the nominal load for which the circuit has been optimized. The gate drivers are basically cascaded buffers. These should be designed in accordance with the rules described in [117] in order to minimize the short-circuit power.

A discussion of other sophisticated optimization techniques such as zero-voltage switching and adaptive dead time control is beyond the scope of this discussion [101].

The efficiency of buck DC-DC converters designed for high output currents can be pushed to values above 90% using the aforementioned design principles [101]. Values of 80% to 90% are state-of-the-art. In the design of DC-DC converters for very low power applications, such high efficiencies are more difficult to achieve. A typical CR16-based IC, e.g. the LmDvp chip described in this chapter, has a power consumption on the order of 100 mW to 200 mW at a supply voltage of 2.5 V in a 0.25 µm CMOS technology. Under these circumstances, the power dissipated in the PWM controller cannot be neglected.

Very high efficiency low output power DC-DC converters can be designed using carefully optimized digital feedback loops based on so-called tapped delay lines. In one published example, an efficiency of 88% was achieved while delivering 5 mW to the load at 1 V in a 0.6 µm CMOS technology [27]. The control circuitry consumed only 10 µW.

The disadvantage of a pure tapped delay line approach is the potentially large area caused by a large number of delay elements in the line and a large multiplexer with one input per delay element. In another published example, a shorter tapped delay line was used in combination with a small fast-clocked counter, so as to reduce the circuit area at the cost of a somewhat larger power consumption [36]. A maximum efficiency of 95% was achieved while delivering 100 mW at 2.5 V in a 0.6 µm CMOS technology. The power consumption of the PWM control circuitry was on the order of 100 µW. It was also shown that similar efficiencies can be achieved for a wide range of output load currents if the channel widths of the power switch transistors are optimized for the expected load current as described above. Even at extremely small load power on the order of hundreds of microwatts, efficiencies of 90% were measured. Another example of high efficiency very low power DC-DC conversion is described in [44]. The PWM control is also based on a modified tapped delay line approach. The circuit was realized in a 0.3 µm CMOS technology. Again, efficiencies of more than 95% were measured in an output power range from 40 mW to 100 mW.

The above arguments clearly show that the need for a second DC-DC converter in DSV circuit and system design does not create any additional power overhead if each converter is carefully optimized for its expected load current. Of course, a second converter takes up large area. This area overhead can be reduced by using a dual output converter. The converter presented in [36] can easily be provided with a second output as described in [28]. The dual output converter has two individually optimized power trains for the two outputs while the complete control circuitry is shared between the two power trains. One output was optimized for 20 mA at 2 V and the other one for 1 mA at 1 V. The measured efficiencies of both outputs were between 80% and 89% over wide ranges of output currents. The efficiency was even pushed towards the 95% margin using a higher quality inductor.

The integration of the converter with the circuits to be supplied on the same chip can further reduce the area. The monolithic integration of the filter components is, however, still difficult. At typical switching frequencies on the order of 1 MHz, the capacitors and particularly the inductors are unfeasibly large. On the other hand, increasing the switching frequency in favour of smaller filter components detracts from the efficiency as mentioned before. Consequently, because of area limitations, the highest efficiency can often not be realized with monolithic integration of the filter components [68]. The single and dual output converters presented in [28], [36], and [44] were all designed as embedded converters with externally connected filter components.

Just recently, a single-inductor dual-output converter has been proposed [72]. In this example, the inductor is shared between two ouputs, which reduces the area overhead. An efficiency of 85% was achieved while delivering 310 mW to the loads. The circuit was realized in a 0.5 µm CMOS technology and the ouput voltages were set to 3 V and 2.5 V.

If several chips in a system are to be supplied with the same voltages, it might still be advantageous to use an external converter shared by all the different chips instead of integrating an embedded converter with every single chip. Since a single converter requires less filter components, the overall area of the system might be smaller. The converter must then, of course, be optimized to the expected total load current.

The above discussion indicates that the generation of a second supply voltage in low power DSV designs can generally be accomplished without introducing additional power overheads. This requires the design of very low power PWM control circuitry and careful optimization of each power train for its respective nominal load current. The inevitable area overhead can be minimized if the converter configuration is carefully optimized according to the specific requirements of the system under consideration.

8.8 Comments

The experiments discussed in this chapter have proven the full compliance of the proposed DSV logic synthesis methodology with existing industrial design flows. The CR16 core module has been chosen as a test case for two reasons. Firstly, it is the central component of numerous embedded microcontroller systems tailored to various types of applications. Secondly, because of the extensive use of clock gating in this module it has been an ideal vehicle for an investigation of the interaction of clock gating and clock voltage scaling.

The different modules of the microcontroller subsystem of the LmDvp chip are all designed in the same way, i.e. using the same synthesis strategies with the same tools in the same environment. Thus, the DSV power optimization method could now be applied to the other modules in a similar manner. The only costly task in this context is the choice of suitable input pattern for each individual module. This requires a profound knowledge of the internal structure and the functionality of the modules. Note that the analysis presented in this chapter has confirmed that insufficient node coverage may result in a significant amount of non-critical cells not being sized appropriately or not being operated at the lower supply voltage, although this has not had a significant impact on the results in the case of the core module. The pattern selection issue is, however, an inherent problem of power-driven logic synthesis in general, rather than a problem of DSV logic synthesis in particular.

The CR16 core has a very limited optimization potential that does not justify the use of DSV logic synthesis for this module. Nevertheless, the methodology can still be useful for optimizating the entire system if, firstly, less critical modules having a larger potential for optimization through DSVS and, secondly, non-critical modules suitable for GSVS, can be identified among the remaining modules. This, of course, requires an in-depth analysis of the characteristics of all modules in the system including the choice of suitable input pattern for the optimization.

Chapter 9

Summary, Conclusions, and Outlook

The main objectives of this study were to implement a new way of optimizing the dynamic power consumption of standard-cell-based ASICs by means of voltage scaling and to investigate the potential and the limitations of this new approach. The methodology developed in this work had to meet three important requirements. Firstly, the power optimization had to be fully automated in order to minimize the additional design time. Secondly, the methodology had to rely solely on standard tools in order to facilitate its integration with existing design flows. Thirdly, no constraints that would prevent standard bulk CMOS fabrication processes from being used were allowed to be introduced.

The discussion of power optimization methods in Chapters 3 and 4 has shown that the majority of practically relevant techniques either exploit the concept of power supply shutdown or optimize switching activities, capacitances, signal transition times, and the channel widths of the transistors. Supply voltage scaling is usually restricted to global strategies that are driven by critical path relaxation through pipelining or parallelization at different levels of abstraction. More advanced supply and threshold voltage scaling approaches have not yet become state-of-the-art.

The voltage regulation techniques discussed in Chapter 4 can be exploited for dynamic or static power optimization or both and are promising regarding ultra-low-voltage operation. The tremendous effort of designing the complex regulation circuitry, however, significantly increases the total design time and cost. Thus, voltage regulation is not suitable for the low to medium volume standard cell ASIC world, where the complete design process is primarily based on HDL modeling and automatic synthesis. In the long term, voltage regulation in ASICs might gain importance if the required circuitry becomes available in the form of pre-designed parameterized building blocks.

In contrast to the voltage regulation techniques, dual supply voltage scaling (DSVS) and dual threshold voltage scaling (DTVS) can be automated in the logic synthesis process and are, thus, well suited for standard-cell-based ASIC design. Both techniques can be exploited for the optimization of the dynamic power consumption. They are applicable to

practically any type of circuit or system and could, in principal, be combined with other advanced voltage scaling techniques in the future. In this study, DSVS has been preferred to DTVS because DSVS is compatible with any conventional bulk CMOS technology.

The results of dynamic power optimization through DSVS published in recent years are promising (see Section 5.3). Logic synthesis, however, is a well-established fully automated task and any new design technique to be used at this stage of the design process must integrate easily with the existing design flows. In other words, it must be supported by the standard synthesis tools. The results published so far were all obtained using special algorithms tailored to DSVS and built into proprietary tools. This is one main reason why DSVS has not yet become an integral part of real-world design flows.

This study shows that DSVS can be carried out exploiting cell-library-based gate sizing algorithms, provided that a suitably modeled dual supply voltage (DSV) standard cell library exists. This does not necessitate special DSVS algorithms or proprietary synthesis tools. The required DSV synthesis library file can easily be created from two conventional SSV libraries. The only costly task remaining is the design of the level-converting flip-flop cells, which, of course, is required by any DSV design methodology. Thus, only little modification of conventional design environments is required for adopting the DSVS concept.

As another important result of the discussion of power optimization techniques in Chapters 3 and 4, a set of state-of-the-art single supply voltage (SSV) power optimization techniques that might come into conflict with or at least have an impact on the effectiveness of DSVS has been identified. The novel DSV logic synthesis methodology proposed in Chapter 6 enables all these techniques to be used simultaneously. Consequently, the results of DSV logic synthesis can always be compared directly with the results of SSV power optimization, which is the only way of revealing the true additional benefit of DSVS. This methodology has been used for investigating the potential and the limitations of DSVS.

The fundamental characteristics of DSVS have been studied on a collection of 48 combinational and sequential MCNC benchmark circuits. For strict and moderately relaxed timing constraints, the power reduction due to DSVS has been up to 20%. Less than 10% power reduction, however, has been observed on average.

A direct comparison with related work requires the selection of circuits, the set of tools, the technology and the library, the supply voltages, and the impact of state-of-the-art power optimization techniques to be taken into account. For this reason, a well-known DSVS algorithm, namely the clustered voltage scaling (CVS) algorithm developed by Usami et al., has been implemented and applied to the combinational benchmark circuits within the existing design environment. On average, only 4% power reduction has been achieved with CVS as opposed to 7% achieved with the novel DSV logic synthesis methodology.

The DSVS technique is generally less effective than claimed by other researchers when it is used in a real-world design environment under realistic conditions. This is primarily due to the smaller optimization potential of the circuits themselves. The analysis presented in Section 7.5.3 has shown that the amount of slack available for exploitation through DSVS

has been much smaller in this work than in related work. In some cases, this discrepancy is primarily due to different strictnesses of the timing constraints. In other cases, the discrepancy can partly be attributed to different characteristics of the timing-driven synthesis; the SIS synthesis package used in related work apparently creates netlists containing fewer timing-critical cells than state-of-the-art synthesis tools, even if the timing constraints are more strict. Finally, the use of state-of-the-art SSV power optimization after the initial timing-driven synthesis has been shown to reduce the amount of slack even further.

A quantification of the optimization potential has been achieved by means of the power savings estimation method (PSEM) proposed in Chapter 6. The results agree well with the actual power savings realized with CVS, which reflects that both algorithms assume an invariant logic structure. The novel DSV logic synthesis methodology yields somewhat better results because logic re-structuring performed in the optimization process increases the optimization potential. While the current implementation works only on combinational circuits, the PSEM could be improved in the future so as to work properly for any design. In that case, it could serve as a tool for predicting the effectiveness of DSVS for specific modules and for optimizing the lower supply voltage before spending the effort of developing a DSV standard cell library.

When the timing constraints are relaxed, the total slack normally increases and the overall effectiveness of DSV power optimization improves. The actual benefit of the individual techniques, however, depends on the circuit structure. Since slightly different timing constraints often lead to very different circuit structures, the effectiveness of DSVS sometimes degrades while that of SSV power optimization improves even more obviously or vice versa as the timing constraints are relaxed. This flexible exploitation of the optimization potential by means of different techniques is an expected and desirable characteristic of the simultaneous use of DSVS and SSV optimization techniques in the DSV logic synthesis.

It has been shown that, on average, the benefit of DSVS grows at an increasing rate as the timing constraints are more and more relaxed. However, when the constraints are relaxed far enough so that global operation of the circuit at the lower supply voltage at the cost of a moderate area overhead is possible, global supply voltage scaling (GSVS) enabled by logic-level parallelization is preferable. This generally restricts the use of DSV logic synthesis to circuits that are subject to the strictest or to moderately relaxed timing constraints.

In order to prove beyond doubt the full compliance of the proposed DSV logic synthesis methodology with existing industrial ASIC design environments, it has been used on NATIONAL SEMICONDUCTOR'S 16-bit CompactRISC processor core module (CR16). The CR16 core is the key component in numerous embedded microcontroller systems. It has been chosen as an example primarily because of the extensive use of clock gating in the design. This makes the CR16 an ideal vehicle for an investigation of the interaction of clock gating and clock voltage scaling.

The CR16 core has turned out to be extremely timing-critical with limited optimization potential even for moderately relaxed timing constraints. Nonetheless, the results of these

experiments prove that DSVS can, in principle, coexist with all common design techniques including the scan test method and clock gating within an industrial design environment.

The proposed methodology supports clock voltage scaling, the primary effect of which is a reduction of the dynamic power consumption in the clock network. However, there are negative and positive secondary effects as well. Level-converting flip-flops introduce extra delay into critical paths, which may degrade the overall circuit performance. The extra delay may also necessitate parallelization or gate up-sizing or both in the combinational parts of the circuit, which creates some power overhead. On the other hand, the effectiveness of DSVS improves because of the large number of level-converting cells that are forced into the design. This leads to additional power savings. Clearly, clock voltage scaling is feasible only if the performance penalty is small and the power overhead is more than compensated by the power reduction in the clock network and the additional power reduction in the logic.

For the CR16 core module, the performance penalty has been negligible because of the relatively long critical paths. However, the power savings have been just large enough to compensate for the power overhead because the clock gating strategy has already very effectively reduced the dynamic power consumption in the clock network. In a typical implementation of the core, the clock network accounts for only 7% of the total dynamic power and, hence, even a significant reduction of the power in the clock network due to clock voltage scaling results in little reduction of the total power. This case study has clearly shown that clock voltage scaling is feasible in general but useful only if clock gating is not possible or if the contribution of the clock network to the total dynamic power consumption of a module is still large in the presence of gated clocks.

A realistic scenario for the application of DSVS, taking into account the characteristics and the limitations of this technique, is as follows. Suppose the design to be optimized is a complex hierarchical system such as the color image processor introduced in Chapter 8. It is composed of numerous modules that are subject to very different timing constraints. A few modules are timing-critical while others are more or less relaxed. When some modules are sufficiently relaxed to be operated completely at the lower one of the two given supply voltages, this can already justify the area overhead and the additional design effort associated with generating the second voltage. Dual supply voltage logic synthesis can then be applied to those of the remaining modules that exhibit optimization potentials large enough to compensate for the overhead caused by the more complex DSV layout. Finally, clock voltage scaling can be applied to modules that fulfill the aforementioned requirements.

In future work, an example of such a positive scenario could be identified and used as a vehicle for a re-investigation of the DSV layout issue including DSV clock network generation, in order to gain a better understanding of the layout-related power overhead and the consequences of increased clock delays. The recent introduction of the first commercial placement tool for DSV layout synthesis will clearly simplify this task. Finally, according to a first analysis, the effectiveness of DSVS is expected to improve in future technology generations as a result of increasing interconnect to device capacitance ratios. This is an interesting aspect and should be investigated in more depth in future work.

Appendix A

Derivation of Consistent Delay, Energy and Power Formulas

The following sections contain derivations of expressions describing the delay of an inverter and the capacitive switching energy. Moreover, a novel expression for the short-circuit power consumption is derived on the basis of the alpha-power-law MOSFET model.

A.1 Inverter Delay

A compact expression for the delay of a CMOS inverter driving an output load capacitance C_{node} can be derived from the alpha-power-law MOSFET model. The delay t_D is the time it takes for the output voltage to reach $V_{DD}/2$ after the input voltage reached the same level.

The derivation is based on the following assumptions and simplifications [95]. Firstly, the input voltage is assumed to rise linearly from 0 V to V_{DD} as shown in Figure A.1. Note that here the input transition time t_T denotes the time it takes for the input voltage to rise all the way from 0 V up to V_{DD}, while often t_T is measured between $V_{in} = 0.1 \cdot V_{DD}$ and $V_{in} = 0.9 \cdot V_{DD}$. Secondly, the input slope is assumed to be at least three times faster than the output slope. Under this condition, according to [95], only the n-channel transistor is relevant and the impact of the p-channel transistor can be neglected.

The horizontal axis in Figure A.1 is divided into four regions (r1). In region one, the input voltage V_{in} is still smaller than the threshold voltage V_t. The n-channel transistor is off and the output voltage is high:

$$V_{out,r1} = V_{DD} \tag{A.1}$$

When V_{in} rises beyond V_t (region two, r2) at t_1, the n-channel transistor enters the saturation region and starts discharging the output capacitor. Thus, the output voltage starts going down. If the input slope is sufficiently fast compared with the output slope, as explained

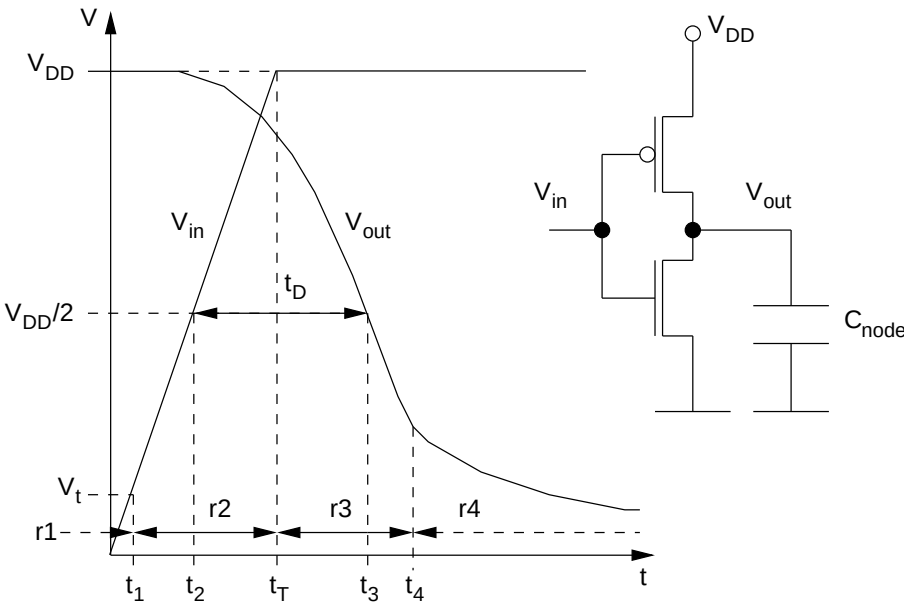

Figure A.1: Voltage waveforms used for calculating the inverter delay.

above, V_{in} reaches V_{DD} before V_{out} reaches $V_{DD}/2$, i.e t_T is smaller than t_3. In region three (r3), V_{in} is fixed at V_{DD} and the n-channel transistor is still in saturation. At the time t_4, the n-channel transistor enters the linear region. In the following derivation, the target time for the delay measurement (t_3), is assumed to fall into region three. If the input slope is very slow, t_3 might fall into region two and the solution becomes complicated. This case is not considered here. For very fast input slopes, t_3 might fall into region four. However, according to [95], the formula derived hereafter is still a good approximation in this case.

In region two, the input voltage V_{in}, which is identical with the gate-source voltage V_{GS} of the n-channel transistor, is

$$V_{in,r2} = \frac{V_{DD}}{t_T} t \quad , \tag{A.2}$$

and the saturated n-channel transistor sinks a current

$$I_{D,r2} = \beta K_{ISAT} \left(\frac{V_{DD}}{t_T} t - V_t \right)^{\alpha} \quad , \tag{A.3}$$

that discharges the capacitor at the output. The output voltage waveform can be obtained solving the following differential equation:

$$C_L \frac{dV_{out,r2}}{dt} = -\beta K_{ISAT} \left(\frac{V_{DD}}{t_T} t - V_t \right)^{\alpha} \tag{A.4}$$

With the initial condition given by Equation A.1, the solution of Equation A.4 is

$$V_{out,r2} = V_{DD} - \frac{\beta K_{ISAT}}{C_L} \frac{1}{\alpha+1} \frac{t_T}{V_{DD}} \left(\frac{V_{DD}}{t_T} t - V_t \right)^{\alpha+1} \quad . \tag{A.5}$$

In region three, the n-channel transistor is still in saturation and V_{in} is fixed at V_{DD}. Hence, the differential equation to be solved is

$$C_L \frac{dV_{out,r3}}{dt} = -\beta K_{ISAT} \left(V_{DD} - V_t\right)^{\alpha} \quad , \tag{A.6}$$

and the solution is given by

$$V_{out,r3} = V_{out,r2}\big|_{t=t_T} - \frac{\beta K_{ISAT}}{C_L} \left(V_{DD} - V_t\right)^{\alpha} \left(t - t_T\right) \quad . \tag{A.7}$$

With Equation A.5, the target time t_3, where V_{out} reaches $V_{DD}/2$, is defined by

$$\frac{V_{DD}}{2} = V_{DD} - \frac{\beta K_{ISAT}}{C_L V_{DD} (\alpha+1)} \left(V_{DD} - V_t\right)^{\alpha+1} t_T - \frac{\beta K_{ISAT}}{C_L} \left(V_{DD} - V_t\right)^{\alpha} \left(t_3 - t_T\right) \quad , \tag{A.8}$$

which leads to

$$t_3 = \frac{C_L}{2\beta K_{ISAT}} \frac{V_{DD}}{\left(V_{DD} - V_t\right)^{\alpha}} + t_T - \frac{1}{\alpha+1} \frac{V_{DD} - V_t}{V_{DD}} t_T \quad . \tag{A.9}$$

Finally, the delay t_D can be calculated from t_3 and $t_2 = t_T/2$:

$$t_D = t_3 - t_2 = \left(\frac{1}{2} - \frac{1 - V_t/V_{DD}}{\alpha+1}\right) t_T + \frac{C_L V_{DD}}{2\beta K_{ISAT} \left(V_{DD} - V_t\right)^{\alpha}} \tag{A.10}$$

The delays for rising and falling input transitions are identical if the inverter is symmetrical.

A.2 Capacitive Switching Energy

When the output voltage of a CMOS inverter rises from 0 V to V_{DD}, as shown in Figure 2.2a, the current i_{cap} charges the capacitor at the output node. Furthermore, if the transition time t_T of the input signal is greater than zero, both transistors are conducting simultaneously for a short period of time, which causes a short-circuit current i_{sc} to flow from the power supply to ground. Assuming that only one such transition occurs for $t > 0$, the total energy drawn from the supply is

$$\begin{aligned} E_{dyn} &= \int_0^{\infty} P_{dyn}(t)\, dt && \text{(A.11)} \\ &= \int_0^{\infty} V_{DD}\, i_{DD}(t)\, dt && \text{(A.12)} \\ &= \int_0^{\infty} V_{DD} \left(i_{sc}(t) + i_{cap}(t)\right) dt && \text{(A.13)} \\ &= \int_0^{\infty} P_{sc}(t)\, dt + E_{cap} \quad , && \text{(A.14)} \end{aligned}$$

where

$$E_{cap} = V_{DD} \int_0^\infty i_{cap}(t)\,dt \tag{A.15}$$

is the energy that is drawn from the supply in order to charge the node capacitance, and P_{sc} is the short-circuit power calculated in the next section.

The current that charges the node capacitance can be written as

$$i_{cap} = C_{node} \frac{dV_{out}(t)}{dt} \quad , \tag{A.16}$$

After inserting Equation A.16, Equation A.15 can be solved:

$$\begin{aligned} E_{cap} &= V_{DD} C_{node} \int_0^\infty \frac{V_{out}(t)}{dt}\,dt && \text{(A.17)} \\ &= V_{DD} C_{node} \left[\lim_{t\to\infty} V_{out}(t) - V_{out}(t=0)\right] && \text{(A.18)} \\ &= V_{DD} C_{node} (V_{DD} - 0) && \text{(A.19)} \\ &= V_{DD}^2 C_{node} && \text{(A.20)} \end{aligned}$$

Note that E_{cap} depends neither on the dimension of the two transistors nor on the input and output waveforms. Equivalent derivations can be found in the literature [5, 17, 121].

The energy stored in the capacitor after completion of the transition is

$$\begin{aligned} E_{cnode} &= \int_0^\infty V_{out}(t)\, i_{cap}(t)\,dt && \text{(A.21)} \\ &= C_{node} \int_0^\infty V_{out}(t) \frac{V_{out}(t)}{dt}\,dt \quad . && \text{(A.22)} \end{aligned}$$

Using the methods of substitution and back substitution, Equation A.22 can be solved:

$$\begin{aligned} E_{cnode} &= C_{node} \frac{1}{2} \left[\lim_{t\to\infty} V_{out}^2(t) - V_{out}^2(t=0)\right] && \text{(A.23)} \\ &= \frac{1}{2} C_{node} (V_{DD}^2 - 0) && \text{(A.24)} \\ &= \frac{1}{2} V_{DD}^2 C_{node} && \text{(A.25)} \end{aligned}$$

One half of E_{cap} is stored in the capacitor while the other half is dissipated in the p-channel transistor. When the output voltage of the inverter falls down to zero again, as shown in Figure 2.2b, a current i_{cap} flows from the capacitor to ground. The capacitor is discharged and E_{cnode} is dissipated in the n-channel transistor.

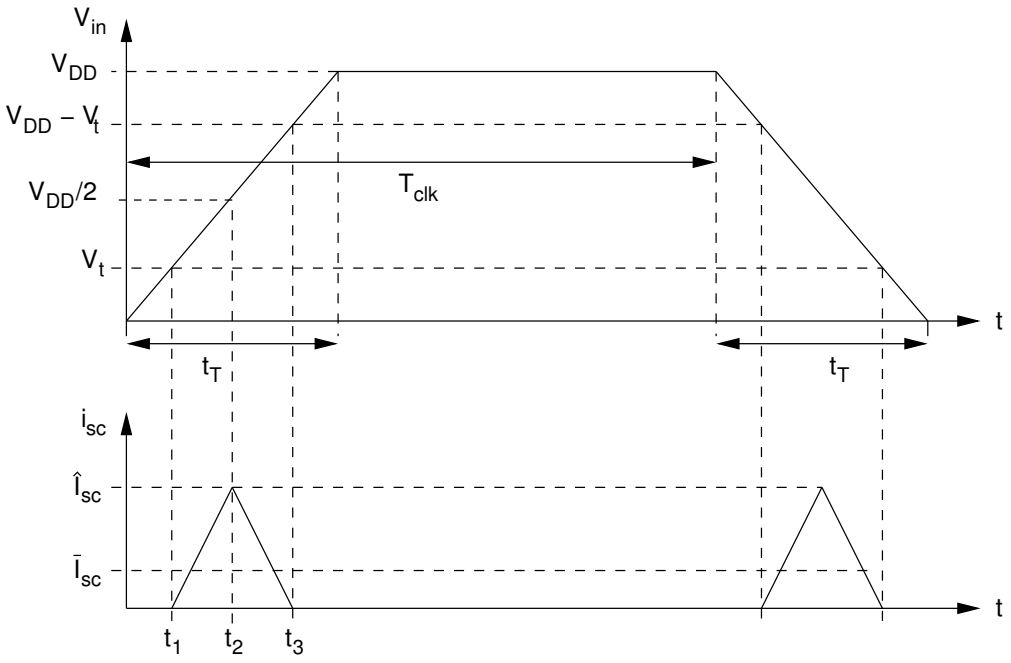

Figure A.2: Triangular approximation of the short-circuit current for an inverter with zero output load capacitance.

A.3 Short-Circuit Power

An expression for the short-circuit power P_{sc} of an inverter can be derived making the following simplifications and assumptions. Firstly, the inverter is symmetrical, i.e. the transconductance β and the threshold voltage V_t are the same for both transistors. Secondly, the capacitance at the output node and, hence, i_{cap} are zero. Thirdly, the short-circuit current i_{sc} is approximated by a triangular waveform as shown in Figure A.2.

The mean short-circuit current $\bar{I}_{sc}$ can be calculated from the waveforms in Figure A.2. Since the inverter is symmetrical, the current waveforms during rising and falling transitions are identical if the input transition time t_T is the same in both cases. Thus, the calculation can be restricted to the rising input transition, where the input voltage is

$$V_{in} = \frac{V_{DD}}{t_T} t \quad . \tag{A.26}$$

The n-channel transistor starts conducting when V_{in} rises beyond V_t at the time

$$t_1 = \frac{V_t}{V_{DD}} t_T \quad , \tag{A.27}$$

the current reaches its maximum $\hat{I}_{sc}$ when V_{in} is equal to $V_{DD}/2$ at the time

$$t_2 = \frac{t_T}{2} \quad , \tag{A.28}$$

and the p-channel transistor stops conducting when V_{in} reaches $V_{DD} - V_t$ at t_3. The current waveform is symmetrical with respect to t_2 and, thus, the calculation can be further restricted to $t_1 \leq t \leq t_2$. During this interval, the n-channel transistor is in saturation and the short-circuit current is

$$i_{sc} = \beta K_{ISAT} \, (V_{in} - V_t)^{\alpha} \quad . \tag{A.29}$$

The short-circuit power can then be calculated as follows:

$$\begin{aligned}
P_{sc} &= V_{DD} \bar{I}_{sc} && \text{(A.30)} \\
&= V_{DD} \frac{2}{T_{clk}} \int_{t_1}^{t_2} i_{sc}(t)\, dt && \text{(A.31)} \\
&= V_{DD} \frac{2}{T_{clk}} \beta K_{ISAT} \int_{t_1}^{t_2} \left(\frac{V_{DD}}{t_T} t - V_t \right)^{\alpha} dt && \text{(A.32)} \\
&= V_{DD} \frac{2}{T_{clk}} \beta K_{ISAT} \left[\frac{t_T}{V_{DD}} \frac{1}{\alpha+1} \left(\frac{V_{DD}}{t_T} t - V_t \right)^{\alpha+1} \right]_{t_1}^{t_2} && \text{(A.33)} \\
&= \frac{2}{(\alpha+1)\, 2^{\alpha+1}} \frac{\beta K_{ISAT}}{T_{clk}} t_T \, (V_{DD} - 2V_t)^{\alpha+1} \quad . && \text{(A.34)}
\end{aligned}$$

Equation A.34 is based on the assumption that one and only one transition, i.e. either a rising or a falling transition, occurs in every clock cycle. This is the largest possible activity if no spurious transitions occur. However, as explained in Section 2.3.1, not every node in a circuit switches in every clock cycle. This can be taken into account by means of the switching activity factor α_{01}. Since α_{01} refers only to rising transitions, an additional factor of two is required when α_{01} is introduced into Equation A.34. Replacing $1/T_{clk}$ with the effective clock frequency $2 \cdot \alpha_{01} \cdot f_{clk}$, the short-circuit power of the inverter becomes

$$P_{sc} = \alpha_{01} \, f_{clk} \frac{4 \beta K_{ISAT}}{(\alpha+1)\, 2^{\alpha+1}} t_T \, (V_{DD} - 2V_t)^{\alpha+1} \quad . \tag{A.35}$$

For α equal to two (no velocity saturation), Equation A.35 has the same characteristics as the expressions frequently found in the literature [117].

Although Equation A.35 has been derived for an inverter, it is often used as an approximation of the short-circuit power consumption of any type of CMOS gate, with β being an effective transconductance parameter representing the driving strength of the gate's symmetrical pull-up and pull-down networks. The total short-circuit power consumption of a circuit can then be calculated by summing Equation A.35 over all gates (see Equation 2.22).

Appendix B

Additional Synthesis Results

The tables presented in this appendix contain results of the optimization of combinational benchmark circuits as a supplement to the in-depth discussion of the characteristics of DSVS that can be found in Section 7.5. These data have not been included directly in the main part of this document since they are not of immediate importance for a basic understanding of the subject. When a deeper understanding of certain experiments and the respective results is required, however, this additional information is valuable. All the tables are referenced from and explained in Section 7.5. Therefore, this appendix contains no further explanations, except for those included in the table captions.

P_{dyn} after ...	SSV pwr. opt.	DSV$^{(*)}$ pwr. opt.	DSVS	CVS
comp. with ...	before pwr. opt.		after SSV pwr. opt.	
alu2	-20%	-21%	-1%	±0%
alu4	-27%	-28%	-3%	-1%
apex6	-26%	-34%	-10%	-8%
apex7	-26%	-32%	-8%	-9%
b9	-23%	-26%	-5%	-3%
c432	-26%	-28%	-3%	±0%
c499	-26%	-28%	-2%	±0%
c880	-22%	-31%	-12%	-4%
c1355	-45%	-45%	±0%	±0%
c1908	-30%	-35%	-7%	-6%
c2670	-28%	-29%	-3%	-2%
c3540	-25%	-29%	-5%	-1%
c5315	-23%	-33%	-12%	-9%
c6288	-20%	-25%	-6%	-2%
c7552	-17%	-24%	-9%	-5%
dalu	-30%	-32%	-3%	±0%
des	-24%	-27%	-6%	-3%
i10	-28%	-38%	-14%	-11%
i5	-26%	-33%	-5%	-6%
lal	-21%	-22%	-2%	-2%
my_adder	-32%	-37%	-13%	-7%
pair	-26%	-33%	-9%	-8%
rot	-31%	-42%	-13%	-12%
term1	-22%	-26%	-5%	-1%
vda	-17%	-18%	-1%	±0%
x1	-25%	-26%	-1%	-2%
x3	-28%	-42%	-20%	-11%
x4	-29%	-36%	-12%	-8%
avg.	-26%	-31%	-7%	-4%

Table B.1: Opt. of comb. benchmarks without area constraints. Critical path delays set to 1.2 times the minimum. $^{(*)}$DSV power opt. includes both DSVS and SSV opt.

	Reduction of P_{dyn} due to CVS applied after ...		
	timing opt.	timing and area opt.	SSV pwr. opt.
alu2	-1%	±0%	±0%
alu4	-7%	-4%	-1%
apex6	-16%	-17%	-8%
apex7	-12%	-13%	-9%
b9	-9%	-8%	-3%
c432	±0%	±0%	±0%
c499	±0%	±0%	±0%
c880	-11%	-9%	-4%
c1355	-1%	±0%	±0%
c1908	-9%	-8%	-6%
c2670	-6%	-2%	-2%
c3540	-5%	-3%	-1%
c5315	-15%	-15%	-9%
c6288	-3%	-4%	-2%
c7552	-10%	-10%	-5%
dalu	-3%	-4%	±0%
des	-16%	-6%	-3%
i10	-14%	-15%	-11%
i5	-13%	-10%	-6%
lal	-3%	-4%	-2%
my_adder	-8%	-9%	-7%
pair	-15%	-15%	-8%
rot	-21%	-17%	-12%
term1	-1%	-1%	-1%
vda	-1%	±0%	±0%
x1	-3%	-3%	-2%
x3	-22%	-22%	-11%
x4	-15%	-11%	-8%
avg.	-8%	-7%	-4%

Table B.2: Optimization of combinational benchmarks using the CVS method. Critical path delays set to 1.2 times the minimum.

P_{dyn} after ...	SSV pwr. opt.	DSV(*) pwr. opt.	DSVS
comp. with ...	before pwr. opt.		after SSV pwr. opt.
c432	-14%	-16%	-2%
c499	-23%	-25%	-2%
c880	-19%	-26%	-8%
c1355	-19%	-20%	-1%
c1908	-21%	-29%	-11%
c2670	-10%	-13%	-3%
c3540	-11%	-16%	-6%
c5315	-16%	-26%	-12%
c6288	-7%	-11%	-5%
c7552	-5%	-11%	-6%
avg.	-16%	-19%	-6%

Table B.3: Optimization of combinational benchmarks. Critical path delays set to 1.1 times the shortest possible critical path delays. (*)DSV power optimization includes both DSVS and SSV techniques.

P_{dyn} after ...	SSV pwr. opt.	DSV(*) pwr. opt.	DSVS
comp. with ...	before pwr. opt.		after SSV pwr. opt.
c432	-9%	-16%	-7%
c499	-27%	-27%	-1%
c880	-24%	-29%	-7%
c1355	-35%	-38%	-4%
c1908	-23%	-32%	-11%
c2670	-28%	-31%	-4%
c3540	-23%	-25%	-3%
c5315	-21%	-34%	-16%
c6288	-15%	-20%	-5%
c7552	-14%	-27%	-15%
avg.	-22%	-28%	-7%

Table B.4: Optimization of combinational benchmarks. Critical path delays set to 1.35 times the shortest possible critical path delays. (*)DSV power optimization includes both DSVS and SSV techniques.

P_{dyn} after ...	SSV pwr. opt.	DSV(*) pwr. opt.	DSVS
comp. with ...	before pwr. opt.		after SSV pwr. opt.
c432	-24%	-29%	-8%
c499	-30%	-30%	-1%
c880	-25%	-34%	-16%
c1355	-38%	-43%	-7%
c1908	-27%	-39%	-17%
c2670	-21%	-24%	-6%
c3540	-25%	-26%	-8%
c5315	-17%	-33%	-21%
c6288	-13%	-17%	-6%
c7552	-19%	-28%	-19%
avg.	-24%	-30%	-11%

Table B.5: Optimization of combinational benchmarks. Critical path delays set to 1.5 times the shortest possible critical path delays. (*)DSV power optimization includes both DSVS and SSV techniques.

	Number of cells (in percent) with...											
	...minimum size				...low supply voltage				...min. size & low volt.			
t_c/t_{cmin}	1.1	1.2	1.35	1.5	1.1	1.2	1.35	1.5	1.1	1.2	1.35	1.5
c432	46	37	52	71	1	8	7	12	1	8	6	10
c499	52	59	89	92	0	0	0	29	0	0	0	26
c880	61	59	73	78	19	29	21	46	18	28	21	46
c1355	72	81	89	94	0	0	19	29	0	0	19	28
c1908	59	72	87	92	19	25	29	36	18	25	28	35
c2670	60	72	81	88	9	9	11	20	9	9	11	19
c3540	63	75	83	85	5	7	6	11	5	7	6	11
c5315	69	78	83	89	38	39	48	56	35	38	47	54
c6288	40	47	57	63	2	3	3	5	2	3	3	4
c7552	66	72	82	86	7	12	22	30	6	12	21	28
avg.	59	65	78	84	10	13	17	27	9	13	16	26

Table B.6: Impact of the delay on the number of cells with minimum size and/or low supply voltage.

	DSVS (2.5 V/1.8 V)		GSVS (2.0 V)	
	P_{dyn}	cell area	P_{dyn}	cell area
	comp. with after SSV pwr. opt. (2.5 V)			
c432	-3%	±0%	-13%	+47%
c499	-2%	±0%	+45%	+52%
c880	-12%	-1%	-20%	+23%
c1355	±0%	+1%	+38%	+32%
c1908	-7%	+1%	+12%	+43%
c2670	-3%	±0%	+10%	+53%
c3540	-5%	-1%	-6%	+31%
c5315	-12%	±0%	-24%	+22%
c6288	-6%	-2%	+33%	+50%
c7552	-9%	-2%	-17%	+28%
avg.	-6%	±0%	+6%	+38%

Table B.7: Comparison of DSVS and GSVS. Critical path delays have been relaxed to 1.2 times the shortest possible critical path delays.

	DSVS (2.5 V/1.8 V)		GSVS (1.8 V)	
	P_{dyn}	cell area	P_{dyn}	cell area
	comp. with after SSV pwr. opt. (2.5 V)			
c432	-7%	-3%	-29%	+52%
c499	-1%	±0%	+38%	+57%
c880	-7%	±0%	-20%	+39%
c1355	-4%	-5%	+17%	+25%
c1908	-11%	+2%	+1%	+52%
c2670	-4%	±0%	+2%	+70%
c3540	-3%	±0%	-12%	+47%
c5315	-16%	-1%	-34%	+26%
c6288	-5%	-2%	+25%	+60%
c7552	-15%	-1%	-29%	+33%
avg.	-7%	-1%	-4%	+46%

Table B.8: Comparison of DSVS and GSVS. Critical path delays have been relaxed to 1.35 times the shortest possible critical path delays.

	DSVS (2.5 V/1.8 V)		GSVS (1.6 V)	
	P_{dyn}	cell area	P_{dyn}	cell area
	comp. with after SSV pwr. opt. (2.5 V)			
c432	-19%	-8%	-40%	+60%
c499	-7%	-3%	+11%	+54%
c880	-16%	+3%	-33%	+48%
c1355	-7%	-4%	+9%	+27%
c1908	-8%	±0%	-27%	+50%
c2670	-6%	±0%	-17%	+72%
c3540	-8%	±0%	-30%	+47%
c5315	-21%	-2%	-48%	+27%
c6288	-6%	+1%	+12%	+68%
c7552	-19%	-1%	-40%	+38%
avg.	-12%	-1%	-20%	+49%

Table B.9: Comparison of DSVS and GSVS. Critical path delays have been relaxed to 1.5 times the shortest possible critical path delays.

	DSVS (2.5 V/1.8 V)		GSVS (2.0 V)	
	P_{dyn}	cell area	P_{dyn}	cell area
	comp. with after SSV pwr. opt. (2.5 V)			
c432	-7%	-3%	-29%	+19%
c499	-1%	±0%	-21%	+5%
c880	-7%	±0%	-19%	+23%
c1355	-4%	-5%	-4%	-1%
c1908	-11%	+2%	-22%	+12%
c2670	-4%	±0%	-15%	+24%
c3540	-3%	±0%	-14%	+20%
c5315	-16%	-1%	-27%	+7%
c6288	-5%	-2%	-20%	+19%
c7552	-15%	-1%	-30%	+11%
avg.	-7%	-1%	-20%	+14%

Table B.10: Results of GSVS (II) strategy, which starts with timing-driven synthesis targeting delays of 1.1 times the minimum. Actual constraints set to 1.35 times the minimum.

	$\Delta P_{dyn}/P_{dyn}$ due to DSVS		Amount of LV cells	
$V_{NWELL,LV}$	V_{DDL}	V_{DD}	V_{DDL}	V_{DD}
alu2	-1%	-1%	$< 1\%$	$< 1\%$
alu4	-3%	-2%	1%	1%
apex6	-10%	-6%	41%	33%
apex7	-8%	-9%	29%	23%
b9	-5%	-2%	29%	19%
c432	-3%	-1%	8%	1%
c499	-2%	-2%	0	0
c880	-12%	-13%	29%	23%
c1355	$\pm$0%	$\pm$0%	0	0
c1908	-7%	-7%	25%	15%
c2670	-3%	-3%	9%	9%
c3540	-5%	-5%	7%	6%
c5315	-12%	-13%	39%	36%
c6288	-6%	-7%	3%	3%
c7552	-9%	-6%	12%	7%
dalu	-3%	-2%	6%	3%
des	-6%	-4%	24%	21%
i10	-14%	-16%	52%	51%
i5	-5%	-3%	40%	21%
lal	-2%	-2%	17%	11%
my_adder	-13%	-7%	23%	10%
pair	-9%	-7%	32%	20%
rot	-13%	-15%	48%	38%
term1	-5%	-2%	6%	3%
vda	-1%	-1%	1%	1%
x1	-1%	$\pm$0%	8%	3%
x3	-20%	-17%	66%	51%
x4	-12%	-12%	27%	20%
avg.	-7%	-6%	21%	15%

Table B.11: Impact of the body effect due to high n-well potential (V_{NWELL}) in low voltage (LV) cells on the effectiveness of DSVS.

Symbols

α	velocity saturation parameter
α_{01}	switching activity factor
β	transconductance parameter
ε_{ox}	dielectric constant of the gate oxide
ε_{si}	dielectric constant of the silicon
Φ_F	difference between Fermi level and intrinsic Fermi level
γ	body factor
μ	carrier mobility
ω	flag that indicates whether a gate can be operated at V_{DDL}
A	area of a library cell
A	total cell area of a circuit
ΔA	deviation of the total cell area of a circuit
A_{LC}	area of a level-converting library cell
A_{min}	smallest possible total cell area of a circuit
A_{seq}	cell area occupied by the sequential parts of a circuit
c	library cell
C_{DB}	drain-to-bulk diffusion capacitance
C_F	filter capacitor
C_G	gate input (gate-to-channel) capacitance
C_{GLC}	gate input capacitance of a level-converting cell
c_{HV}	high voltage library cell

C_{int}	interconnect/wire capacitance
C_L	parasitic capacitance of a filter inductor
c_{LC}	level-converting library cell
c_{LV}	low voltage library cell
$C(n)$	set of library cells that implement the functionality $F(n)$
C_{node}	node/load capacitance
$C_{node,max}$	maximum output load capacitance used for cell characterization
$C_{node,min}$	minimum output load capacitance used for cell characterization
c_{opt}	library cell that implements $F(n)$ while minimizing $COST$
$COST$	cost of a particular implementation of a logic network NW
C_{ox}	gate oxide capacitance
D	depletion layer thickness
D	duty cycle of v_{rec}
$DeltaCOST$	reduction of $COST$ due to substitution of a cell
D_H	Hamming distance
E	switching energy
E_{intF}	cell-internal switching energy for falling input edge
E_{intR}	cell-internal switching energy for rising input edge
E_{totF}	total switching energy for falling input edge
E_{totR}	total switching energy for rising input edge
F	functionality of a library cell c
f_{clk}	clock frequency
$F(n)$	required functionality of node n in a logic network NW
FSL	fan-out signal level property of a library cell
f_{sw}	frequency of v_{rec}
f_{vco}	frequency of the signal generated by a VCO
i,k	general purpose indices and variables
i_{cap}	capacitive switching current

I_D	drain current
I_{DSUB}	drain current in the subthreshold regime
I_{DSUB0}	drain current at V_{GS} equal to V_t
i_L	inductor current
I_{load}	load current
$I_{quiescent}$	quiescent current
i_{sc}	short-circuit current
ISL	input signal level property of a library cell
k,i	general purpose indices and variables
K,M,N	general purpose variables
K_{ILIN}	fitting parameter for the drain current in the linear region
K_{ISAT}	saturation current fitting parameter
$k(m)$	number of zero-to-one transitions in m clock cycles
K_{SUB}	subthreshold current fitting parameter
K_{VSAT}	saturation voltage fitting parameter
L	gate length
L_F	filter inductor
m	counter for the number of clock cycles
M,N,K	general purpose variables
n	process parameter
n	node in a logic network NW
N,K,M	general purpose variables
N_A	doping concentration in p-type material
N_D	doping concentration in n-type material
n_i	intrinsic carrier density
NW	logic network
p	parameter describing the delay increment due to voltage scaling
P	dynamic power consumption of a library cell

P_{cap}	capacitive switching power
ΔP_{cap}	deviation of the capacitive switching power
P_{clk}	dynamic power consumption of the clock network
ΔP_{clk}	deviation of the dynamic power in the clock network
P_{comb}	dynamic power consumption of the combinational parts of a circuit
ΔP_{comb}	deviation of the dynamic power in the combinational parts
P_{dyn}	dynamic power consumption
ΔP_{dyn}	deviation of the dynamic power consumption
P_{HV}	dynamic power consumption of a high voltage library cell
P_{LC}	dynamic power consumption of a level-converting library cell
P_{LV}	dynamic power consumption of a low voltage library cell
psc	fitting parameter describing the contribution of P_{sc} to P_{dyn}
P_{sc}	short-circuit power
P_{seq}	dynamic power consumption of the sequential parts of a circuit
ΔP_{seq}	deviation of the dynamic power in the sequential parts
P_{stat}	static power consumption
PSX	power savings index
P_T	probability of a state transition to occur
P_{tot}	total power consumption
P_{vdd}	dynamic power consumption of a gate supplied with V_{DD}
S_t	reciprocal subthreshold slope
t_c	critical path delay
T_{clk}	clock period
T_{cmin}	shortest possible critical path delay
$t_{control}$	duration of one control time step
t_D	gate delay
Δt_D	deviation of the gate delay
t_{DHV}	high voltage gate delay

t_{DLC}	delay of a level-converting gate
t_{DLV}	low voltage gate delay
t_{DTE}	delay to endpoint
Δt_{DTE}	deviation of the delay to endpoint
$t_{iterate}$	duration of one iteration period
t_{max}	largest acceptable path delay
t_{ox}	gate oxide thickness
t_{path}	path delay
t_Q	clock-to-output delay
t_{QHV}	clock-to-output delay of a high voltage flip-flop
t_{QLC}	clock-to-output delay of a level-converting flip-flop
t_{QLV}	clock-to-output delay of a low voltage flip-flop
t_{setup}	setup time
$t_{setupHV}$	setup time of a high voltage flip-flop
$t_{setupLC}$	setup time of a level-converting flip-flop
$t_{setupLV}$	setup time of a low voltage flip-flop
T_{sw}	period of v_{rec}
t_T	(input) signal transition time
t_{Teq}	equivalent low voltage input signal transition time
$t_{T,max}$	maximum input signal transition time used for cell characterization
$t_{T,min}$	minimum input signal transition time used for cell characterization
t_{TO}	output signal transition time
t_{TOHV}	output signal transition time of a high voltage cell
t_{TOLC}	output signal transition time of a level-converting cell
t_{TOLV}	output signal transition time of a low voltage cell
V_{DD}	(nominal/high) supply voltage
V_{DDL}	low supply voltage
V_{DDp}	supply voltage value that leads to $1+p$ times larger delays

V_{DS}	drain-source voltage
V_{DSN}	drain-source voltage of n-channel transistor
V_{DSSAT}	saturation voltage
V_{FB}	flat band voltage
V_{GS}	gate-source voltage
V_{in}	input voltage
V_{leak}	output voltage of leakage sensor
V_{node}	node voltage
V_{out}	output voltage
v_{rec}	PWM signal in DC-DC converter
V_{SB}	source-bulk voltage
V_{sub}	substrate voltage
$V_{sub,n}$	substrate voltage for n-channel transistors
$V_{sub,p}$	substrate voltage for p-channel transistors
V_t	threshold voltage
V_{t0}	threshold voltage for zero source-bulk voltage
$V_{t0,max}$	max. threshold voltage for zero source-bulk voltage
$V_{t0,min}$	min. threshold voltage for zero source-bulk voltage
V_{TH}	thermal voltage
V_{thigh}	high threshold voltage
q	elementary charge
W	gate width

Abbreviations and Acronyms

A	input pin
ACPI	Advanced Configuration and Power Interface
alu,ALU	arithmetic logic unit
AND	AND gate
APM	Advanced Power Management
ASIC	application specific integrated circuit
BIC	bus inversion coding
BIU	bus interface unit
bsh	barrel shifter
bsm	bus state machine
BUF	buffer cell
BUFLC	level-converting buffer cell
CAR	compare address register
CBC	core bus controller
CCIR	Consultative Committee for International Radio
CFG	configuration register
CISC	complex instruction set computer
CLK	clock (signal / input pin)
CLR	clear input pin
CMOS	complementary metal oxide semiconductor
CMOS9	NATIONAL SEMICONDUCTOR'S 0.18 µm CMOS technology

CMOSX-9	library in CMOS9 technology
CPU	central processing unit
CR16	NATIONAL SEMICONDUCTOR'S 16-bit CompactRISC processor core
CSU	chip select unit
CVS	clustered voltage scaling
D	data input pin
dbg	debug module
DBS	debug base register
DC	direct current
DC	don't care
DCR	debug control register
DECT	Digital Enhanced Cordless Telephone
DFFQ	D-flip-flop cell with Q output only
DFFQLC	level-converting D-flip-flop cell with Q output only
DFG	data flow graph
DMA	direct memory access
dp,DP	data path
DPM	dynamic power management
DSP	digital signal processor
DSR	debug status register
DSV	dual supply voltage
DSVL018	DSV library in 0.18 µm CMOS
DSVL025	DSV library in 0.25 µm CMOS
DSVS	dual supply voltage scaling
DTV	dual threshold voltage
DTVS	dual threshold voltage scaling
DVPO	dual voltage power optimization
ECVS	extended clustered voltage scaling

EDIF	Electronic Design Interchange Format
EN	enable signal
esm	execution state machine
FIFO	first-in-first-out buffer
FF	flip-flop
FSM	finite state machine
GND	ground
GPIO	general purpose I/O
Gscale	name of a particular DSVS algorithm
GSVS	global supply voltage scaling
HCMOS7	STMICROELECTRONICS' 0.25 µm CMOS technology
HDL	hardware description language
HV	high voltage
I2C	Inter Integrated Circuit (bus)
IC	integrated circuit
INV	inverter cell
INVLC	level-converting inverter cell
I/O	input/output
IP	intellectual property
ISCAS	International Symposium on Circuits and Systems
ISP	interrupt stack pointer
ITU-R	International Telecommunication Union Radiocommunication Sector
JPEG	Joint Photographic Experts Group
LC	level converter
LmDvp	NATIONAL SEMICONDUCTOR'S digital color image processor
LUT	look-up table
LV	low voltage
MCNC	Microelectronics Center North Carolina

MOS	metal oxide semiconductor
MOSFET	metal oxide semiconductor field effect transistor
MPEG	Moving Picture Experts Group
MPEG-4	video coding standard defined by the MPEG
MSV	multiple supply voltage
MTCMOS	multiple threshold voltage CMOS
mul	multiplier
MUX	multiplexer
MWIS	maximum-weighted independent set
NAND	NAND gate
NOR	NOR gate
NTSC	National Television Systems Committee
NVB	NATIONAL SEMICONDUCTOR'S video bus
OR	OR gate
PAL	Phase Alternation by Line
PBC	peripheral bus controller
PBM	probability based mapping
pc,PC	program counter
PDP	power-delay product
PERL	Practical Extraction and Report Language
PLD	programmable logic device
PLL	phase locked loop
PMC	power-manageable component
PMP	power management unit
PMU	power manager unit
PREZ	preset input pin
PSEM	power savings estimation method
PSR	processor status register

PTL	pass transistor logic
PWM	pulse width modulation
Q	data output pin
QN	inverting data output pin
qu	decode and displacement unit (queue)
r,R	register
RA	return address pointer
RAM	random access memory
rf	register file
RISC	reduced instruction set computer
ROM	read only memory
RTL	register transfer level
SD	scan data input pin
SDFFCP	scan-D-flip-flop cell with clear and preset inputs
SDFFCPLC	level-converting scan-D-flip-flop cell with clear and preset inputs
SDI	serial debug interface
SE	scan enable input
SIS	system for sequential circuit synthesis
SO	scan data output pin
SoC	system on a chip
SOI	silicon on insulator
SP	stack pointer
SP	standard products
SPI	Serial Peripheral Interface
SSV	single supply voltage
TCB	time-critical boundary
TCL	Tool Command Language
TV	television

UART	Universal Asynchronous Receiver/Transmitter
USART	Universal Synchronous/Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
USP	user stack pointer
VCO	voltage controlled oscillator
VHDL	Very High Speed Integrated Circuit Hardware Description Language
XOR	XOR gate
XNOR	XNOR gate
Z	output pin
ZN	inverting output pin
ZDVD	zero delay virtual driver cell

Bibliography

[1] B. Ackalloor and D. Gaitonde, ”An overview of library characterization in semi-custom design,” *IEEE Custom Integrated Circuits Conference*, 1998, pp. 305–312.

[2] R. Altherr, ”Reduktion der Verlustleistung von ICs durch eine chipintegrierte Versorgungsspannungsregelung mit einem Boost-Konverter,” *Dissertation (in German)*, University of Ulm, Germany, 2002.

[3] A. Alvandpour and C. Svensson, ”Improving cell libraries for low power design,” *Proc. Int. Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 1996, pp. 317–325.

[4] A. Alvandpour, P. Larsson-Edefors, and C. Svensson, ”Separation and extraction of short-circuit power consumption in digital CMOS VLSI circuits,” *Proc. Int. Symp. Low Power Electronics and Design*, 1998, pp. 245–249.

[5] A. Bellaouar and M. Elmasry, *Low-Power Digital VLSI Design*, Kluwer Academic Publishers, Boston, 1995.

[6] L. Benini and G. De Micheli, *Dynamic Power Management*, Kluwer Academic Publishers, Boston, 1998.

[7] L. Benini and G. De Micheli, ”System-level power optimization: techniques and tools,” *Proc. Int. Symp. Low Power Electronics and Design*, 1999, pp. 288–293.

[8] L. Benini, A. Bogliolo, and G. De Micheli, ”A survey of design techniques for system-level dynamic power management,” *IEEE Trans. VLSI Systems*, vol. 8, no. 3, 2000, pp. 299–316.

[9] R. Brayton, G. Hachtel, and A. Sangiovanni-Vincentelli, ”Multilevel logic synthesis,” *Proc. of the IEEE*, vol. 78, no. 2, 1990, pp. 264–299.

[10] J. Berthold, R. Nadal, and C. Heer, ”Optionen fuer Low-Power-Konzepte in den sub-180-nm-CMOS-Technologien,” *U.R.S.I. Kleinheubacher Tagung (solicited talk)*, Miltenberg, Germany, 2002.

[11] L. Benini, G. De Micheli, and E. Macii, ”Designing low-power circuits: practical recipes,” *IEEE Circuits and Systems Magazine*, vol. 1, no. 1, 2001, pp. 6–25.

[12] M. Borah, R. M. Owens, and M. J. Irwin, ”Transistor sizing for minimizing power consumption of CMOS circuits under delay constraints,” *Proc. Int. Symp. Low Power Design*, 1995, pp. 167–172.

[13] K. Bowman et al., ”A physical alpha-power law MOSFET model,” *Proc. Int. Symp. Low Power Electronics and Design*, 1999, pp. 218–222.

[14] T. Burd et al., ”A dynamic voltage scaled microprocessor system,” *Proc. IEEE Int. Solid-State Circuits Conf.*, 2000, pp. 294–295.

[15] T. Callaway, ”Modeling the power consumption of CMOS arithmetic elements,” *Application Specific Processors*, E. Swartzlander (ed.), Kluwer Academic Publishers, Boston, 1997, pp. 29–61.

[16] T. Callaway and E. Swartzlander, ”The power consumption of CMOS adders and multipliers,” *Low-Power CMOS Design*, A. Chandrakasan and R. Brodersen (eds.), IEEE Press, Piscataway, 1998, pp. 218–224.

[17] A. Chandrakasan and R. Brodersen, *Low Power Digital CMOS Design*, Kluwer Academic Publishers, Boston, 1995.

[18] A. Chandrakasan et al., ”Optimizing power using transformations,” *IEEE Trans. Computer Aided Design of Integrated Circuits and Systems*, vol. 14, no. 1, 1995, pp. 12–31.

[19] C. Chen and M. Sarrafzadeh, ”Power reduction by simultaneous voltage scaling and gate sizing,” *Proc. Asia and South Pacific Design Automation Conference*, 2000, pp. 333–338.

[20] C. Chen, A. Srivastava, and M. Sarrafzadeh, ”On gate level power optimization using dual-supply voltages,” *IEEE Trans. VLSI Systems*, vol. 9, no. 5, 2001, pp. 616–629.

[21] B. Chen and I. Nedelchev, ”Power Compiler: a gate-level power optimization and synthesis system,” *Proc. IEEE Int. Conf. Computer Design*, 1997, pp. 74–79.

[22] J.-M. Chang and M. Pedram, ”Energy minimization using multiple supply voltages,” *IEEE Trans. VLSI Systems*, vol. 5, no. 4, 1997, pp. 436–443.

[23] Collaborative Benchmarking Laboratory (CBL), LGSynth93 benchmark set used in conjunction with 1993 MCNC Int. Workshop on Logic Synthesis, *http://www.cbl.ncsu.edu*, 2002.

[24] J. Cong, L. He, and C.-K. Koh, "Layout optimization," *Low Power Design in Deep Submicron Electronics*, W. Nebel and J. Mermet (eds.), Kluwer Academic Publishers, Dordrecht, 1997, pp. 205–265.

[25] O. Coudert, "Gate sizing for constrained delay/power/area optimization," *IEEE Trans. VLSI Systems*, vol. 5, no. 4, 1997, pp. 465–472.

[26] S. Cristoloveanu, "Silicon on insulator technology," *The VLSI Handbook*, W.-K. Chen (ed.), CRC Press, Boca Raton, 2000, pp. 4/1–4/15.

[27] A. Dancy and A. Chandrakasan, "Ultra low-power control circuits for PWM converters," *Proc. IEEE Power Electronics Specialists Conf.*, 1997, pp. 21–27.

[28] A. Dancy, R. Amirtharajah, and A. Chandrakasan, "High-efficiency multiple-output DC-DC conversion for low-voltage systems," *IEEE Trans. VLSI Systems*, vol. 8, no. 3, 2000, pp. 252–263.

[29] N. Dragone et al., "An innovative methodology for the design automation of low power libraries," *Proc. Int. Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 1998, pp. 31–40.

[30] D. Frank et al., "Supply and threshold voltage optimization for low power design," *Proc. Int. Symp. Low Power Electronics and Design*, 1997, pp. 317–322.

[31] D. Frank, "Application and technology forecast," *Low Power Design in Deep Submicron Electronics*, W. Nebel and J. Mermet (eds.), Kluwer Academic Publishers, Dordrecht, 1997, pp. 9–44.

[32] J. Frenkil, "Tools and methodologies for low power design," *Proc. Design Automation Conf.*, 1997, pp. 76–81.

[33] D. Galbi, "Understanding your cell library: cell characterization for the masses," *Boston Synopsys Users Group Meeting*, Boston, 1999.

[34] S. Gary, "Low-power microprocessor design," *Low Power Design Methodologies*, J. Rabaey and M. Pedram (eds.), Kluwer Academic Publishers, Boston, 1996, pp. 255–288.

[35] R. Gonzales, B. Gordon, and M. Horowitz, "Supply and threshold voltage scaling for low power CMOS," *IEEE J. Solid-State Circuits*, vol. 32, no. 8, 1997, pp. 1220–1216.

[36] J. Goodman, A. Dancy, and A. Chandrakasan, "An energy/security scalable encryption processor using an embedded variable voltage DC/DC converter," *IEEE J. Solid-State Circuits*, vol. 33, no. 11, 1998, pp. 1799–1809.

[37] S. Gupta and R. Gupta, "ASIC design," *The VLSI Handbook*, W.-K. Chen (ed.), CRC Press, Boca Raton, 2000, pp. 64/1–64/28.

[38] V. Gutnik and A. Chandrakasan, "Variable-voltage digital signal processing," *Low-Power CMOS Design*, A. Chandrakasan and R. Brodersen (eds.), IEEE Press, Piscataway, 1998, pp. 166–176.

[39] G. Hachtel and F. Somenzi, *Logic Synthesis and Verification Algorithms*, Kluwer Academic Publishers, Boston, 1996.

[40] M. Hamada et al., "A top-down low power design technique using clustered voltage scaling with variable supply-voltage scheme," *Proc. IEEE Custom Integrated Circuits Conference*, 1998, pp. 495–498.

[41] , "Cascode voltage switch logic: a differential CMOS logic family," *Proc. IEEE Int. Solid-State Circuits Conference*, 1984, pp. 16–17.

[42] M. Hirabayashi, K. Nose, and T. Sakurai, "Design methodology and optimization strategy for dual-V_{TH} scheme using commercially available tools," *Proc. Int. Symp. Low Power Electronics and Design*, 2001, pp. 283–286.

[43] D. Hodges and H. Jackson, *Analysis and Design of Digital Integrated Circuits*, 2nd ed., McGraw-Hill, New York, 1988.

[44] F. Ichiba et al., "Variable supply-voltage scheme with 95%-efficiency DC-DC converter for MPEG-4 codec," *Proc. Int. Symp. Low Power Electronics and Design*, 1999, pp. 54–59.

[45] IC Insights, Inc., *The McClean Report*, Scottsdale, 2001.

[46] IC Insights, Inc., *The McClean Report*, Scottsdale, 2002.

[47] M. Igarashi et al., "A low-power design method using multiple supply voltages," *Proc. Int. Symp. Low Power Electronics and Design*, 1997, pp. 36–41.

[48] H. Igura et al., "An 800-MOPS, 110-mW, 1.5-V, parallel DSP for mobile multimedia processing," *IEEE J. Solid-State Circuits*, vol. 33, no. 11, 1998, pp. 1820–1828.

[49] S. Iman and M. Pedram, *Logic synthesis for low power VLSI design*, Kluwer Academic Publishers, Boston, 1998.

[50] K. Itoh, "Low power memory design," *Low Power Design Methodologies*, J. Rabaey and M. Pedram (eds.), Kluwer Academic Publishers, Boston, 1996, pp. 201–251.

[51] M. Izumikawa et al., "A 0.25-µm CMOS 0.9-V 100-MHz DSP core," *IEEE J. Solid-State Circuits*, vol. 32, no. 1, 1997, pp. 52–61.

[52] M. Johnson and K. Roy, ”Datapath scheduling with multiple supply voltages and level converters,” *ACM Trans. Design Automation of Electronic Systems*, vol. 2, no. 3, 1997, pp. 227–248.

[53] V. v. Kaenel, P. Macken, and M. Degrauwe, ”A Voltage reduction technique for battery-operated systems,” *IEEE J. Solid-State Circuits*, vol. 25, no. 5, 1990, pp. 1136–1140.

[54] V. v. Kaenel et al., ”Automatic adjustment of threshold and supply voltages for minimum power consumption in CMOS digital circuits,” *Proc. IEEE Symp. Low Power Electronics*, 1994, pp. 78–79.

[55] S.-M. Kang and Y. Leblebici, *CMOS Digital Integrated Circuits Analysis and Design*, 2nd ed., McGraw-Hill, Boston, 1999.

[56] J. Kao, A. Chandrakasan, and D. Antoniadis, ”Transistor sizing issues and tool for multi-threshold CMOS technology,” *Proc. Design Automation Conf.*, 1997, pp. 409–414.

[57] J. Kao, S. Narendra, and A. Chandrakasan, ”MTCMOS hierarchical sizing based on mutual exclusive discharge patterns,” *Proc. Design Automation Conf.*, 1998, pp. 495–500.

[58] Y. Katsumata et al., ”CMOS/BiCMOS technology,” *The VLSI Handbook*, W.-K. Chen (ed.), CRC Press, Boca Raton, 2000, pp. 2/1–2/28.

[59] K. Keutzer and P. Vanbekbergen, ”The impact of CAD on the design of low power digital circuits,” *Proc. IEEE Symp. Low Power Electronics*, 1994, pp. 42–45.

[60] M. Khellah and M. Elmasry, ”Power minimization of high-performance submicron CMOS circuits using a dual-V_{dd} dual-V_{th} (DVDV) approach,” *Proc. Int. Symp. Low Power Electronics and Design*, 1999, pp. 106–108.

[61] M. Kontiala, M. Kuulusa, and J. Nurmi, ”Comparison of static logic styles for low-voltage digital design,” *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, 2001, pp. 1421–1424.

[62] C. Kretzschmar, R. Siegmund, and D. Mueller, ”A low overhead auto-optimizing bus encoding scheme for low power data transmission,” *Proc. Int. Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2002, pp. 342–352.

[63] H. Kubosawa et al., ”A 1.2-W, 2.16-GOPS/720-MFLOPS embedded superscalar microprocessor for multimedia applications,” *IEEE J. Solid-State Circuits*, vol. 33, no. 11, 1998, pp. 1640–1647.

[64] T. Kuroda et al., "A 0.9-V, 150-MHz, 10-mW, 4 mm^2, 2-D discrete cosine transform core processor with variable threshold-voltage (VT) scheme," *IEEE J. Solid-State Circuits*, vol. 31, no. 11, 1996, pp. 1770–1779.

[65] T. Kuroda and T. Sakurai, "Threshold-voltage control schemes through substrate-bias for low-power high-speed CMOS LSI design," *J. VLSI Signal Processing Systems*, vol. 13, no. 2/3, 1996, pp. 191–201.

[66] T. Kuroda et al., "Variable supply-voltage scheme for low-power high-speed CMOS digital design," *IEEE J. Solid-State Circuits*, vol. 33, no. 3, 1998, pp. 454–462.

[67] T. Kuroda and M. Hamada, "Low-power CMOS digital design with dual embedded adaptive power supplies," *IEEE J. Solid-State Circuits*, vol. 35, no. 4, 2000, pp. 652–655.

[68] V. Kursun et al., "Efficiency analysis of a high frequency buck converter for on-chip integration with a dual-V_{DD} microprocessor," *Proc. European Solid-State Circuits Conf.*, 2002, pp. 743–746.

[69] W. Lee et al., "A 1-V programmable DSP for wireless communications," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, 1997, pp. 1766–1776.

[70] Y.-R. Lin, C.-T. Hwang and A. Wu, "Scheduling techniques for variable voltage low power designs," *ACM Trans. Design Automation of Electronic Systems*, vol. 2, no. 2, 1997, pp. 81–97.

[71] D. Liu and C. Swensson, "Trading speed for low power by choice of supply and threshold voltages," *IEEE J. Solid-State Circuits*, vol. 28, no. 1, 1993, pp. 10–17.

[72] D. Ma, W.-H. Ki, and C.-Y. Tsui, "A pseudo-CCM/DCM SIMO switching converter with freewheel switching," *Proc. IEEE Int. Solid-State Circuits Conf.*, 2002, pp. 390–391.

[73] T. Mahnke et al., "Power optimization through dual supply voltage scaling using power compiler," *European Synopsys Users Group Meeting*, Paris, 2002.

[74] T. Mahnke et al., "Optimizing power using advanced voltage scaling techniques in logic synthesis," *Proc. Int. Conf. VLSI*, 2002, pp. 45–49.

[75] T. Mahnke et al., "Dual supply voltage scaling in a conventional power-driven logic synthesis environment," *Proc. Int. Workshop on Power and Timing Modeling, Optimization and Simulation (PATMOS)*, 2002, pp. 146–155.

[76] T. Mahnke et al., "Efficiency of dual supply voltage logic synthesis for low power in consideration of varying delay constraint strictness," *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, 2002, pp. 701–704.

[77] T. Mahnke et al., ”Impact of technology evolution on dual supply voltage scaling and gate resizing in power-driven logic synthesis,” *Proc. IEEE Int. Conf. Electronics, Circuits and Systems*, 2002, pp. 697–700.

[78] T. Mahnke et al., ”Exploration of dual supply voltage logic synthesis in state-of-the-art ASIC design flows,” *Advances in Radio Science*, published online at www.copernicus.org, 2003.

[79] J. Meindl, ”Low power microelectronics: retrospect and prospect,” *Proc. IEEE*, vol. 83, no. 4, 1995, pp. 619–635.

[80] G. De Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw-Hill, New York, 1994.

[81] M. Miyazaki, H. Mizuno, and K. Ishibashi, ”A delay distribution squeezing scheme with speed-adaptive threshold-voltage CMOS (SA-Vt CMOS) for low voltage LSIs,” *Proc. Int. Symp. Low Power Electronics and Design*, 1998, pp. 48–53.

[82] J. Monteiro and S. Devadas, ”Techniques for power estimation and optimization at the logic level: a survey,” *J. VLSI Signal Processing Systems*, vol. 13, no. 2/3, 1996, pp. 259–276.

[83] S. Mutoh et al., ”1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS,” *IEEE J. Solid-State Circuits*, vol. 30, no. 8, 1995, pp. 847–854.

[84] S. Mutoh et al., ”A 1-V multithreshold-voltage CMOS digital signal processor for mobile phone application,” *IEEE J. Solid-State Circuits*, vol. 31, no. 11, 1995, pp. 1795–1802.

[85] S. Narendra, D. Antoniadis, and V. De, ”Impact of using adaptive body bias to compensate die-to-die Vt variation on within-die Vt variation,” *Proc. Int. Symp. Low Power Electronics and Design*, 1999, pp. 229–232.

[86] L. Nielsen et al., ”Low-power operation using self-timed circuits and adaptive scaling of the supply voltage,” *IEEE Trans. VLSI Systems*, vol. 2, no. 4, 1994, pp. 391–397.

[87] T. Nishikawa et al., ”A 60 MHz 240 mW MPEG-4 video-phone LSI with 16 Mb embedded DRAM,” *Proc. IEEE Int. Solid-State Circuits Conf.*, 2000, pp. 230–231.

[88] R. Panda and F. N. Najm., ”Technology-dependent transformations for low-power synthesis,” *Proc. Int. Conf. Computer Aided Design*, 1997, pp. 650–655.

[89] R. Paul, *MOS-Feldeffekt Transistoren*, Springer-Verlag, Berlin, 1994.

[90] T. Pering, T. Burd, and R. Brodersen, ”Voltage scheduling in the lpARM microprocessor system,” *Proc. Int. Symp. Low Power Electronics and Design*, 2000, pp. 96–101.

[91] C. Piguet et al., ”Low-power design of 8-b embedded CoolRisc microcontroller cores,” *IEEE J. Solid-State Circuits*, vol. 32, no. 7, 1997, pp. 1067–1077.

[92] F. Pollack, ”New microprocessor challenges in the coming generations of CMOS technologies,” *Symp. on Microarchitecture (solicited talk)*, Haifa, Israel, 1999.

[93] D. Pradhan et al., ”Gate-level synthesis for low-power using new transformations,” *Proc. Int. Symp. Low Power Electronics and Design*, 1996, pp. 297–300.

[94] S. Raje and M. Sarrafzadeh, ”Scheduling with multiple supply voltages,” *Integration — The VLSI Journal*, vol. 23, 1997, pp. 37–59.

[95] T. Sakurai and R. Newton, ”Alpha-power law MOSFET model and its application to CMOS inverter delay and other formulas,” *IEEE J. Solid-State Circuits*, vol. 25, no. 2, 1990, pp. 584–594.

[96] T. Sakurai, H. Kawaguchi, and T. Kuroda, ”Low-power CMOS design through V_{TH} control and low-swing circuits,” *Proc. Int. Symp. Low Power Electronics and Design*, 1997, pp. 1–6.

[97] S. Shigematsu et al., ”A 1-V high-speed MTCMOS circuit scheme for power-down application circuits,” *IEEE J. Solid-State Circuits*, vol. 32, no. 6, 1997, pp. 861–869.

[98] SIA Semiconductor Industry Association, *National Technology Roadmap for Semiconductors - Technology Needs*, 1997.

[99] M. Stan and W. Burleson, ”Bus-invert coding for low-power I/O,” *IEEE Trans. VLSI Systems*, vol. 3, no. 1, 1995, pp. 49–58.

[100] M. Stan and W. Burleson, ”Low-power encodings for global communication in CMOS VLSI,” *IEEE Trans. VLSI Systems*, vol. 5, no. 4, 1997, pp. 444–455.

[101] A. Stratakos et al., ”High-efficiency low-voltage DC-DC conversion for portable applications,” *Low-Voltage/Low-Power Integrated Circuits and Systems*, E. Sánchez-Sinencio and A. Andreou (eds.), IEEE Press, Piscataway, 1999, pp. 361–397.

[102] S.-W. Sun and P. Tsui, ”Limitation of CMOS supply-voltage scaling by MOSFET threshold-voltage variation,” *IEEE J. Solid-State Circuits*, vol. 30, no. 6, 1995, pp. 947–949.

[103] V. Sundararajan and K. Parhi, ”Low power synthesis of dual threshold voltage CMOS VLSI circuits,” *Proc. Int. Symp. Low Power Electronics and Design*, 1999, pp. 139–144.

[104] K. Suzuki et al., "A 300 MIPS/W RISC core processor with variable supply-voltage scheme in variable threshold-voltage CMOS," *IEEE Custom Integrated Circuits Conference*, 1997, pp. 587–590.

[105] C. Svensson and J. Yuan, "Latches and flip-flops for low-power systems," *Low-Power CMOS Design*, A. Chandrakasan and R. Brodersen (eds.), IEEE Press, Piscataway, 1996, pp. 233–238.

[106] C. Svennson and D. Liu, "Low power circuit techniques," *Low Power Design Methodologies*, J. Rabaey and M. Pedram (eds.), Kluwer Academic Publishers, Boston, 1996, pp. 37–63.

[107] C. Svennson, "Low voltage technologies," *Low Power Design in Deep Submicron Electronics*, W. Nebel and J. Mermet (eds.), Kluwer Academic Publishers, Dordrecht, 1997, pp. 493–509.

[108] R. Swansson and J. Meindl, "Ion-implanted complementary MOS transistors in low voltage circuits," *IEEE J. Solid-State Circuits*, vol. 7, 1972, pp. 146–153.

[109] Synopsys Inc., "Leakage power optimization using power compiler and multi-threshold CMOS technologies," *European Synopsys Users Group Meeting (tutorial)*, Munich, 2003.

[110] S. Sze, *Physics of Semiconductor Devices*, 2nd ed., John Wiley & Sons, New York, 1981.

[111] M. Takahashi, "A 60-mW MPEG4 video codec using clustered voltage scaling with variable supply-voltage scheme," *IEEE J. Solid-State Circuits*, vol. 33, no. 11, 1998, pp. 1772–1780.

[112] V. Tiwari et al., "Reducing power in high-performance microprocessors," *Proc. Design Automation Conf.*, 1998, pp. 732–737.

[113] K. Usami and M. Horowitz, "Clustered voltage scaling technique for low-power design," *Proc. Int. Symp. Low Power Design*, 1995, pp. 3–8.

[114] K. Usami et al., "Automated low-power technique exploiting multiple supply voltages applied to a media processor," *Proc. IEEE Custom Integrated Circuits Conference*, 1997, pp. 131–134.

[115] K. Usami et al., "Automated low-power technique exploiting multiple supply voltages applied to a media processor," *IEEE J. Solid-State Circuits*, vol. 33, no. 3, 1998, pp. 463–472.

[116] K. Usami et al., "Design methodology of ultra low-power MPEG4 codec core exploiting voltage scaling techniques," *Proc. Design Automation Conf.*, 1998, pp. 483–488.

[117] H. Veendrick, ”Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits,” *IEEE J. Solid-State Circuits*, vol. 19, 1984, pp. 468–473.

[118] J.-S. Wang et al., ”Design of standard cells used in low-power ASIC’s exploiting the multiple-supply-voltage scheme,” *Proc. IEEE ASIC Conf.*, 1998, pp. 119–123.

[119] L. Wei et al., ”Design and optimization of dual-threshold circuits for low-voltage low-power applications,” *IEEE Trans. VLSI Systems*, vol. 7, no. 1, 1999, pp. 16–24.

[120] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design*, 2nd ed., Kluwer Academic Publishers, Boston, 1994.

[121] G. Yeap, *Practical Low Power Digital VLSI Design*, Kluwer Academic Publishers, Boston, 1998.

[122] C. Yeh et al., ”Gate-level design exploiting dual supply voltages for power-driven applications,” *Proc. Design Automation Conf.*, 1999, pp. 68–71.

[123] C. Yeh et al., ”Layout techniques supporting the use of dual supply voltages for cell-based designs,” *Proc. Design Automation Conf.*, 1999, pp. 62–67.

[124] C. Yeh et al., ”Power reduction through iterative gate sizing and voltage scaling,” *Proc. IEEE Int. Symp. Circuits and Systems*, 1999, vol. 1, pp. 246–249.

[125] C. Yeh and Y.-S. Kang, ”Cell-based layout techniques supporting gate-level voltage scaling for low power,” *IEEE Trans. VLSI Systems*, vol. 8, no. 5, 2000, pp. 62–67.

[126] R. Zimmermann and W. Fichtner, ”Low-power logic styles: CMOS versus pass-transistor logic,” *IEEE J. Solid-State Circuits*, vol. 32, no. 7, 1997, pp. 1079–1090.