INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Low-Power Standard Cell Library for Synthesis

by

Ronny Hirsch, B. Sc.

This thesis is submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Engineering

Ottawa-Carleton Institute of Electrical Engineering

Department of Electronics

Carleton University

Ottawa, Canada

September, 1995

© Copyright 1995, Ronny Hirsch

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-26983-3

The undersigned recommend to the Faculty of Graduate Studies
and Research acceptance of this thesis

“Low-Power Standard Cell Library
for Synthesis”

submitted by Ronny Hirsch, B. Sc.
in partial fulfillment of the requirements for
the degree of Master of Engineering

Thesis Supervisor

Chair, Department of Electronics

Carleton University

September 1995

Abstract

The realization of deep sub-micron technologies and the increase in the density of Integrated Circuits (ICs) have made power consumption a major concern in VLSI design. Advanced IC design methodologies employ automatic synthesis tools in conjunction with standard cell libraries to implement digital circuits. In this thesis, a low power cell library is developed, with the objective of minimizing the power dissipation of synthesized circuits. The thesis contains an analysis of the power and speed characteristics of different size cells, and presents a technique that allows to trade speed for power without compromising performance requirements. An experimental infrastructure which determines the power consumption of relatively large circuits has been created to evaluate the quality of the library. Three benchmark designs are used to illustrate the performance of several versions of the library (in terms of power dissipation), and simulation results predict up to a 30% improvement in the power consumption of designs mapped to the proposed library.

Acknowledgements

I would like to express my grateful appreciation to my supervisor, Prof. Martin Lefebvre, for his guidance and on going support. His continuous encouragement and valuable advice made this research possible. I would also like to thank him for providing me the unique opportunity of studying in Canada.

My sincere thanks to Prof. Garry Tarr, Prof. R. Harrison and Ms. Angela Zelmer for their time spent helping me, as a foreign student, on different matters. Thanks are also due to the office staff, Nagui Mikhail, Barbara Lynn, Dennis Piamonte, Betty Zahalan and Alana Wirta, for being so friendly and helpful. Thanks to David Skoll and Arthur Castonguay for helping me in CAD related issues. This research would not be possible without the financial support of the Natural Sciences and Engineering Research Council of Canada, Micronet, and the Department of Electronics of Carleton University.

I would like to thank the cell design group at Northern Telecom, for assisting me in carrying out my experiments. In particular, I wish to thank Trevor Monson for his expert advice and commitment, Ivan Martin for his tremendous support, and Rob Lemieux for keeping my workstation running. Thanks to Neil Pickles, Minh Phan and Barry Rezansoff for “sharing” their CPUs with me during my extensive simulations. Special thanks to Ronald Alleyne, who helped me with PowerMill and Vertue. Dana Coombs and David Choue from EPIC Designs provided excellent product support (PowerMill) and a special evaluation license for this research.

My special thanks to Tal Lichtenstein and her family, for welcoming me into their home and hearts, and for treating me as a member of their family.

Finally, I would like to thank my dear family in Israel, whose complete support and faith kept me going.

Table of Contents

Chapter 1	Introduction	1
1.1	Perspective	1
1.2	Objectives	2
1.3	Thesis Outline	2
Chapter 2	Background - Digital Design for Low-Power	4
2.1	Introduction	4
2.2	The Sources of Power Dissipation in CMOS ICs	4
2.3	Low Power Design Methodologies	5
2.3.1	Circuit/Logic Level Techniques	6
2.3.1.1	Supply Voltage Reduction	6
2.3.1.2	Physical Capacitance Reduction	8
2.3.1.3	Choice of Logic Style	9
2.3.1.4	Complex Gates	12
2.3.2	Architecture Level Techniques	12
2.3.3	Technology and Process Enhancements	13
2.3.4	Other Low Power Techniques	14
2.4	Summary	14
Chapter 3	Multiple Drive, Low-Power Standard Cell Library	15
3.1	Introduction	15
3.2	HDL Synthesis Process	16
3.3	Multiple Drive Cells	18
3.3.1	Drive Capability	18
3.3.2	Cell Utilization During Technology Mapping	18
3.3.3	Power and Delay Characteristics	19
3.4	The “kcell” Library	22
3.4.1	Technology	23
3.4.2	Logic Style	23
3.4.3	Transistor Sizing	23
3.4.3.1	Single Stage Cells	23
3.4.3.2	Multiple Stage Cells	25
3.4.4	Performance	25

3.5	Modifying the “kcell” Library	26
3.5.1	New Drive Strengths: Lower than X1	26
3.5.2	New Drive Strengths: Between X1 and X2	26
3.6	Summary	28
Chapter 4	The Implementation of the Multiple Drive Library	29
4.1	Design Considerations	29
4.2	Transistor Sizing	30
4.3	Layout Format.....	31
4.3.1	Cell Dimensions and Topology.....	31
4.3.2	Layer Constraints.....	33
4.4	Library Development Phases	33
4.4.1	Physical Layouts	34
4.4.2	Cell Characterization	34
4.4.3	Synopsys Library	36
4.4.3.1	Technology Library.....	37
4.4.3.2	Symbol Library	37
4.4.4	Verilog Library.....	38
4.4.5	EPIC Library.....	39
4.5	Additional Library Versions.....	39
4.5.1	The “kcell.p3” and “kcell.p4” Versions	40
Chapter 5	Experimental Procedure.....	42
5.1	Introduction.....	42
5.2	Synthesis and Power Simulation Methodology	43
5.2.1	Synthesis Environment	44
5.2.2	Power Simulation Environment.....	44
5.3	Preparing PowerMill for Simulation.....	45
5.3.1	PowerMill’s Features and Capabilities	45
5.3.2	EPIC/BATMOS Technology File	45
5.3.3	Translating the Synthesis Result to an “Epic” Netlist.....	46
5.4	Statistical Wire Load Models.....	47
5.5	Benchmark Designs	48
5.5.1	Data-Path.....	48
5.5.2	Data-Path Control Unit (DCU) - B62	48
5.5.3	Programmable Line Card Controller - A34	49

Chapter 6	Experimental Results	50
6.1	Simulation and Synthesis Results: Small Data-Path	50
6.1.1	Wire Load Model - SMALL	50
6.1.2	Wire Load Model - LARGE	52
6.2	Simulation and Synthesis Results: DCU (B62)	54
6.2.1	Wire Load Model - MEDIUM	55
6.2.2	Wire Load Model - LARGE	57
6.3	Simulation and Synthesis Results: A34	59
6.4	The Synthesis Results of the “kcell.p3” and “kcell.p4” Library Versions	61
6.4.1	Mapping to “kcell.p3”	61
6.4.2	Mapping to “kcell.p4”	62
6.5	Summary	63
Chapter 7	Conclusions	64
7.1	Summary	64
7.2	Contributions	65
7.3	Future Research	65
	Glossary of Terms	67
	List of Symbols	70
	Appendix A: Multiple Drive Library Listing	71
	Appendix B: Synopsys Library Models	75
	Appendix C: The Verilog Description of the Data-Path	78
	References	80

List of Figures

Figure 2.1	Hierarchical Design Space of Digital ICs.....	6
Figure 2.2	Delay and Power-Delay Product of an Inverter.....	7
Figure 2.3	Conventional CMOS Complex Gate - AOI 32	12
Figure 3.1	Basic Digital IC Design Flow	16
Figure 3.2	Short Circuit Current for Non-Ideal Input Signal	20
Figure 3.3	Average Power [wt] vs. Load Cap. [F] for Different Inverter Instances.....	21
Figure 3.4	Delay [S] vs. Load Cap. [F] - Inverter Cell Instances	22
Figure 3.5	Defining the Minimum Gate Width	24
Figure 3.6	Average Power [wt] vs Load Cap. [F] - Nand2 Cell Instances.....	27
Figure 4.1	Experimental Procedure to Determine Best Rise/Fall Times	31
Figure 4.2	Layout Format for Standard Cell Library	32
Figure 4.3	Library Development Process.....	34
Figure 4.4	Measurement Points for Timing Characterization	35
Figure 4.5	The Role of Library Compiler	36
Figure 4.6	Synopsys Technology Library Structure.....	38
Figure 5.1	Synthesis and Power Simulation Methodology	43
Figure 5.2	Basic Data Path Unit.....	48

List of Tables

Table 3.1	Transistor's Region of Operation, Between t_1 and t_2	20
Table 3.2	Fixed Input Capacitance and Transistor Widths	25
Table 4.1	Input Capacitance and Transistor Widths	30
Table 4.2	"Area" Attribute Modification Factors - kcell.p2 Library	40
Table 5.1	PowerMill vs. HSPICE Simulation Results.....	46
Table 6.1	Power Consumption of the Various Data-Path Implementations (Wire Load Model: SMALL).....	51
Table 6.2	The Drive Utilization of DP_A.S.....	52
Table 6.3	Power Consumption of the Various Data-Path Implementations (Wire Load Model: LARGE	53
Table 6.4	The Drive Utilization of DP_A.L	54
Table 6.5	Power Consumption of the Various DCU Implementations (Wire Load Model: MEDIUM).....	55
Table 6.6	The Drive Utilization of DCU_A.M and DCU_B.M	57
Table 6.7	Power Consumption of the Various DCU Implementations (Wire Load Model: LARGE)	58
Table 6.8	The Drive Utilization of DCU_A.L and DCU_B.L.....	59
Table 6.9	Power Consumption of the A34.....	60
Table 6.10	Drive Utilization - A34 (MEDIUM and LARGE wire load models	61
Table 6.11	Comparison of Drive Utilization: The A34 Mapped to Different Target Libraries (Using the LARGE wire load model).....	62

Chapter 1

Introduction

1.1 Perspective

In recent years, power consumption/dissipation has become one of the most limiting factors in the design of electronic systems. The quality and the cost of products, like laptop and notebook computers, cellular phones and other battery operated systems are defined by features like size, weight, and battery life. The power requirements of such products have a direct impact on those features. Even when power is available (in non-portable applications), the demand for low power is prompted by considerations such as low cost packaging and adequate cooling for the high density integrated circuits (ICs).

As digital VLSI circuits are broadly used in the above described applications, there has been a growing research effort in developing methodologies and techniques that minimize the power requirements of the ICs. In particular, cell libraries are the building blocks of any semi custom digital IC, and as such, have a great impact on the overall power dissipation. Therefore, special attention to the low power issues at this level results in significant power saving.

This thesis addresses the problem of generating a low power standard cell library which primarily serves as a target library for synthesis tools. The cells are modified in

such way that optimal instance utilization will take place during synthesis, to yield a low power implementation. Three different size designs have been used as benchmarks for testing the proposed library and comparing the performance of several versions.

1.2 Objectives

There are two main objectives in this thesis: The first, is to investigate the merits of multiple drive cells in reducing the power of synthesized circuits. For this purpose, the “kcell” library developed at NT/BNR has been taken as a reference, and a new library containing 100 new cells has been implemented. The second objective is to put in place an experimental infrastructure to allow accurate simulation of power in relatively large circuits. It is essential for evaluating the performance of the cell libraries (in terms of power).

1.3 Thesis Outline

Chapter 2 presents the fundamental concepts of digital design for low power in CMOS VLSI circuits. The focus is on the most prominent circuit and logic level techniques that minimize power. Other related hierarchical methods are discussed as well.

Chapter 3 contains several issues related to multiple drive cells: It provides the necessary background for the understanding of HDL synthesis, and the way in which library cells are selected during technology mapping. The power and delay characteristics of multiple cell instances are analyzed, both theoretically and experimentally. A thorough description of the “kcell” library follows, including low power design considerations. Finally, a proposal is made to further improve the “kcell” library by adding new drive versions.

Chapter 4 describes the design and implementation phases of the multiple drive cell library which has been developed for this thesis. It describes the characterization

process as well as the various library models required for the integration of several CAD tools.

Chapter 5 provides details on the experimental infrastructure and procedures of the thesis. It describes the integration of PowerMill and Vertue into the synthesis and simulation environments, as well as the benchmark designs that were used for comparison purposes. Two of the benchmarks have been designed at NT, and are protected by proprietary agreement, thus only the required information is presented in this text.

Chapter 6 contains the simulation results and the cell utilization analysis of the benchmark designs. It includes comparison between the results obtained for the various library versions.

Chapter 7 concludes the thesis and offers recommendations for further research.

Chapter 2

Background - Digital Design for Low Power

2.1 Introduction

According to the many papers and research results published thus far, it is quite clear that the issue of low power should be approached as a multi-level problem and addressed throughout all design phases. Many techniques that minimize power exist at different levels of the design hierarchy. For a given design, only a combination of such methods results in low power implementation. This chapter provides an overview on the most prominent techniques that have proven to be efficient in decreasing power in digital CMOS ICs. The focus however, is on circuit and logic level techniques.

2.2 The Sources of Power Dissipation in CMOS ICs

Power dissipation in CMOS digital ICs arises from two different mechanisms: **dynamic power** which results from switching capacitive loads between two different voltage states, and **static power**, which results from resistive paths to ground. Equation 2.1 represents all the elements involved in CMOS power dissipation [1]:

$$P_{Total} = p_t (C_L \cdot V \cdot V_{dd} \cdot f_{CLK}) + (I_{sc} \cdot V_{dd}) + (I_{Leakage} \cdot V_{dd}) \quad (2.1)$$

The dynamic power is comprised of the first and second terms, whereas the static power is represented by the third term. The first term represents the switching component, where C_L is the loading capacitance, f_{CLK} is the clock frequency, and p_t is the probability that a power consuming transition occurs (the activity factor). In most cases, the voltage swing V is the same as the supply voltage V_{dd} , however, there are cases where the voltage swing on internal nodes may be less than V_{dd} , especially in pass - transistor implementations. The second term is caused by the direct path short circuit current I_{sc} , which occurs when both the NMOS and PMOS transistors are simultaneously active, conducting direct current from source to ground. The third term, caused by the leakage currents $I_{Leakage}$, occurs due to drain junction leakage and subthreshold effects. This current is determined by technology and fabrication considerations. In a “properly designed” circuit, the dominant term is the switching component, thus most of the effort in reducing the power at the circuit level concentrates on minimizing V_{dd} , C_L , f_{clk} , and p_t [1, 2, 3].

The amount of energy required to charge and/or discharge a given load capacitance during each transition is known as the “power-delay” product [1] and is often used as a comparison measure to determine the “quality” of a design with respect to power. Assuming that most of the power is dissipated due to the first term in equation 2.1, the “power-delay” product is given by equation 2.2 [1]:

$$Energy/Transition = \frac{P_{Total}}{f_{CLK}} = C_{effective} \cdot V_{dd}^2 \quad (2.2)$$

where $C_{effective}$ is the effective capacitance being switched and is given by $C_{effective} = p_t \cdot C_L$

2.3 Low Power Design Methodologies

In order to minimize power in digital ICs, low power design techniques should be implemented at each level of the design hierarchy (Figure 2.1) [30]. Different techniques

can be used at each level, and the choice between the options depends on the application. The focus of this section is on techniques implemented at the “Circuit” level, which mostly affect the performance of the library cells. “Architecture” techniques and technology enhancements are important measures that compensate for the increased delays in the circuits (due to supply voltage reduction) [1, 2, 3], and will be described in this context.

Figure 2.1: Hierarchical Design Space of Digital ICs [30]

2.3.1 Circuit/Logic Level Techniques

2.3.1.1 Supply Voltage Reduction

According to equations 2.1 and 2.2 it is evident that scaling down the supply voltage yields the largest reduction of power, and hence is the key for low power operation (because of the quadratic dependence on V_{dd}). However, there is a speed penalty associated with reducing V_{dd} , especially when its value approaches the sum of threshold voltage of the devices. Equation 2.3 [1] further demonstrates this by presenting the first order derivation of the delay of a CMOS gate (long channel) driving a fixed capacitive load C_L as a function of V_{dd} . For V_{dd} values much greater than V_t , the latter can be ignored and the delay is inversely proportional to the supply voltage. As V_{dd} approaches V_t , the denominator decreases and the delay rapidly increases.

$$T_d = \frac{C_L \times V_{dd}}{I} = \frac{C_L \times V_{dd}}{\mu C_{ox} (w/L) (V_{dd} - V_t)^2} \quad (\text{long channel}) \quad (2.3)$$

For deep sub-micron processes, the expression for the current (I) in equation 2.3 is not valid since the saturation drain current I_{Dsat} is limited by velocity saturation of the carriers, and roughly is a linear function of the gate voltage ($I_{Dsat} \sim k \cdot [V_{dd} - V_t]$) [1], as opposed to the squared function in equation 2.3. I_{Dsat} is therefore reduced when the supply voltage is lowered, but the voltage to which circuit capacitance must be charged is reduced by almost the same factor (deep sub micron processes). Thus reducing V_{dd} has a relatively small effect on the switching speed and delays.

Figure 2.2 shows the HSPICE simulation results for the propagation delay and power-delay product of an inverter driving two inverters of the same size as a function of supply voltage V_{dd} . The level 3 HSPICE model which takes into consideration the short channel effects has been used for the simulation.

Figure 2.2: Delay and Power-Delay Product of an Inverter

It can be seen that the power-delay product can be drastically reduced by down scaling V_{dd} from 5V to 3V (2.2.b), which results in a relatively small increase of the propagation delay (2.2.a). For V_{dd} values lower than 3V, the exercise is not beneficial since the delay rapidly increases with only a moderate decrease of the power-delay product.

Another conclusion from similar experiments using various technologies [1], is that the power-delay product improves as delays increase and therefore it is desirable to operate at the slowest possible speed. Since the objective is to minimize power while maintaining the computational throughput, compensation is required for the increased delays, and some of the techniques presented in this chapter were especially developed for this purpose.

2.3.1.2 Physical Capacitance Reduction

Since the power dissipation is approximately a linear function of the capacitance (equation 2.1) it is necessary to reduce the overall capacitance of a design layout as much as possible. Considering a CMOS logic gate, its capacitance at the output is the sum of three components: $C_{INT-Diff}$, C_{WIRE} , and C_{LOAD} [29]. $C_{INT-Diff}$ represents the internal capacitance of the gate which largely consists of the diffusion capacitance of the drain. C_{WIRE} is the interconnect capacitance between the logic gates, and C_{LOAD} represents the sum of gate capacitance of the transistors fed by the output.

All three components need to be minimized in order to save power. The majority of power is dissipated by switching gate capacitance (C_{LOAD}). This component can be effectively reduced by using minimum size transistors since the gate capacitance is proportional to $w \cdot L$. However, it results in speed degradation due to the reduction of charging/discharging current (proportional to $\frac{W}{L}$, whether the device is velocity saturated or not).

Mathematical optimization techniques are often used to implement circuits with optimal transistor sizes by creating cost functions for speed, area and power ([9], [10]). These solutions provide a trade-off between speed and power, depending on the constraints.

The realization of deep sub micron technologies creates a reality where the interconnect capacitances become more dominant than the other two, and a rule of thumb for 0.5 μm technologies is that 60% of the power is dissipated by the interconnects. The reduction of these capacitances depend on the quality of the “place and route” and layout floorplanning CAD tools. The power has been reduced by up to 20% when using floorplanning tools that have cost functions for power, as part of the optimization algorithms [5].

2.3.1.3 Choice of Logic Style

There are various topology and circuit design approaches to implement a given logic and arithmetic function. The choice between these styles is usually subject to criteria such as speed, ease of design and testability, rather than just power dissipation [1]. The “best” logic family for implementing a given function with specified timing constraints, is one that minimizes the power-delay product [30]. The following is a brief summary of the trade-offs with respect to power for some of the well known logic families.

A. Dynamic vs. Static Logic

In terms of low power, it seems that dynamic logic has prominent advantages over static logic in the following areas [1]:

1. Spurious Transitions: In a static implementation, a node can have multiple transitions before settling to the correct logic level. These spurious transitions dissipate extra power over that strictly required to perform the computation. Although it is possible to eliminate most of these transitions with careful logic design, dynamic logic does not have this problem at all, since any node has at most one power consuming transition per clock cycle.

2. Short Circuit Currents: Direct path short circuit currents (second term in equation 2.1) are found in static CMOS circuits, as opposed to dynamic logic where these currents do not occur, except for those cases in which static pull-up transistors are used to compensate for charge sharing problems.

3. Parasitic Capacitance: Since dynamic logic typically uses fewer transistors to implement a given logic function, the total amount of capacitance being switched is much lower, thereby reducing the power and power-delay product (equations 2.1 & 2.2).

4. Switching Activity: This is the only area in which static logic has advantage over dynamic logic since for the latter, each node has to be precharged in every clock cycle. In some cases, nodes are precharged only to be immediately discharged during the evaluation phases, resulting in a higher activity factor that causes additional power dissipation. Furthermore, the clock buffers that drive the precharge transistors also consume extra power.

B. Conventional Static vs. Pass Gate Logic

The complementary pass gate logic (CPL) family is attractive for low power operation since substantially fewer transistors are required to implement important logic functions such as XORs and FFs, which are the building blocks of most arithmetic functions [1, 14, 29]. This allows multipliers and adders to be implemented with a minimal number of transistors. The main problem with this family is the threshold voltage drop across a single pass transistor which results in a reduced current drive and a slower operation at low voltage. Scaling down the threshold voltage has proven to be an effective way to solve this problem, yet for deep sub-micron technologies, there is a limit on the maximal reduction since it may result in subthreshold leakage and diminished noise margins if taken too far.

C. Synchronous vs. Self-Timed Logic

In synchronous designs, there is a continuous switching activity in logic blocks between registers, thus power-down techniques are required to limit the ineffective

switching of nodes. These techniques need to be realized by special circuitry which “detects” whether a specific functional block must or must not operate at a given time. Internal clocks are provided only for those blocks that perform “useful” operation at that time [13, 14, 24, 27]. Major power saving can take place by using power-down strategies, yet, these require additional design effort. On the other hand, self-timed logic is “by definition” a power-down mode for unused blocks, since transitions occur only when requested. The main problem with self-timed logic is that it needs the generation of complementary signals to indicate whether the outputs of logic modules are valid. It has been found [1] that in some cases self-timed implementations can prove to be expensive in terms of energy, especially for data-paths that are continuously computing.

2.3.1.4 Complex Gates

Relatively simple logic functions can be implemented by using complex gates (Figure 2.3) rather than standard basic gates (AND, OR, INV). The advantage of using these cells, is that less transistors are required, and many nodes and interconnect wires are “eliminated”. Hence, the switching capacitance, as well as the activity factor are substantially reduced [4]. Complex gates are usually included in target cell libraries for synthesis tools, and were found to be very useful during technology decomposition and mapping [5, 7, 8, 11]. Power savings of 20% [5] and 50% [11] were reported. The problem with complex gates is that in many cases, the realization of the logic functions that they implement requires transistor branches (of the same type) to be connected in series. This results in speed degradation which causes the automatic synthesis tools to ignore them during technology mapping.

Figure 2.3: Conventional CMOS Complex Gate - AOI 32 ($f = \overline{ABC + DE}$)

2.3.2 Architecture Level Techniques

For the implementation of a low-power design, “Architecture” level techniques (Figure 2.1) are often used in conjunction with “Circuit” techniques. The main purpose of the “Architecture” techniques is to compensate for the reduced circuit speed caused by the down scaling of supply voltage. Prominent techniques are parallelism, pipelining and a combination of parallelism + pipelining [1, 3]. The experimental results of an 8-bit adder [1], demonstrate the effectiveness of these techniques: Initially, one computational unit has been used to implement the adder, with a supply voltage of 5V. In the subsequent experiments, the supply voltage was scaled down to 2.9V. In the second experiment, two identical units were used to implement the same functionality, but each unit worked at half the original frequency while maintaining the throughput. The exercise yielded a decrease of 64% of the power, at the expense of doubling the area and the capacitance in compari-

son to the original implementation. In the third experiment, a pipeline implementation of the data-path was used, which resulted in a 61% reduction of power with only a 15% increase of the capacitance. The combination of both techniques diminished the power by 80%, but again, increased the area and capacitance by a factor of 2.5.

The significant power savings in these experiments had been obtained since the modifications in the architecture allowed a reduction of the speed requirements, and hence the supply voltage could be lowered from 5V to 2.9V.

2.3.3 Technology and Process Enhancements

Scaling down the technology parameters greatly improves the power-delay product since it allows the reduction of supply voltage without increasing delays: In sub-micron technologies when the carriers are velocity saturated, the driving currents are almost linear with the supply voltage and the delays are nearly independent of V_{dd} (Equation 2.3). Ideal scaling [3] means the reduction of all feature sizes by a constant scale factor γ ($\gamma < 1$), including the voltage and all the linear dimensions. This yields [3] a decrease of γ^3 in the energy per operation, and γ^4 reduction of the power-delay product.

In most cases, “ideal” scaling is not performed since the threshold voltage is the limiting factor in this respect. The limit is set by the requirement to retain adequate noise margins and to avoid an increase in the subthreshold leakage currents [1]. In addition, the optimal supply voltage for a deep sub-micron technology takes into account reliability considerations such as hot carriers (caused by high electric fields) which may lead to electromigration problems. A study on 0.35 μm and 0.25 μm technologies [18] examined these issues and suggested various supply voltage levels for various threshold voltages. The main conclusion is that even if other than “ideal” technology scaling is performed, downsizing and other process improvements at this level result in major power saving [14, 15, 17, 19].

2.3.4 Other Low Power Techniques

Although not covered in this text, there are many other methods that save power. In particular, decisions taken at the early stages of the design (“System” and “Algorithm” levels) have a great impact on the power dissipation of the final implementation. For example, a wide range of transformations can be done at the behavioural description of a design. The goal is to reduce the number of cycles in a computation and/or decrease the number of resources for the computation [5]. In this context, there is a growing effort to develop and implement high level synthesis (HLS) techniques that use cost functions for power, and implement a specific design based on its power constraints [22, 23, 25].

2.4 Summary

There is a large variety of techniques that reduce power in digital ICs, and the effectiveness of each method depends on the application. It is important to keep in mind that the best way to implement a low power design is to approach the problem at all design levels, and minimize the components of equation 2.1. As seen in this chapter, architecture, circuit, and technology level techniques are closely related. In reality, their implementation in a given design often results in major power saving while maintaining the computational throughput. Trade-offs between various circuit level techniques were explained as well. These concepts and considerations were extremely helpful for the design and implementation of the low power cell library that was developed for this thesis.

Chapter 3

Multiple Drive, Low Power

Standard Cell Library

3.1 Introduction

The realization of a low power standard cell library requires each cell to be designed for minimum power. Some of the techniques and considerations presented in Chapter 2 are appropriate for this purpose. However, these measures alone are insufficient since most of the advanced digital design methodologies include automatic synthesis tools as part of the design flow. Thus, the power dissipation of synthesized circuits is not only determined by the quality of the library cells, but also by the ability of the synthesis tool to generate a low power implementation [5, 7]. This chapter explains the major issues associated with synthesis, with an emphasis on those related to the cell library and the selection of cells. In addition, it provides an overview on the power and delay characteristics of multiple drive cells, and the possible benefits of using such cells within a target library. Finally, this chapter introduces the “kcell” library which is the reference and benchmark for this thesis.

3.2 HDL Synthesis Process

Hardware description languages (HDLs) describe the architecture and behavior of discrete electronic systems, and play an important role in modern IC design methodologies. Figure 3.1 shows a basic design flow that includes a synthesis tool and a logic simulator.

Figure 3.1: Basic Digital IC Design Flow

This digital IC design flow is typical for most automatic synthesis tools, and in this work, it has been realized with Verilog (HDL), and Synopsys (Synthesis Tool). Therefore, the specific details discussed here are related to Synopsys and Verilog, yet the main ideas are general, and valid for other tools and languages as well.

The process of converting an HDL description to a gate level implementation is known as “Logic Synthesis”, and three major steps are associated with the synthesis and optimization process:

1) Flattening - is a logic optimization step that removes all intermediate variables and uses boolean distributive laws to remove all parentheses. Thus, flattening removes all the logic structure from a design. It is a way of eliminating inefficient structure.

2) Structuring - refers to factorization. Structure is added to a design by factoring out common sub-expressions as intermediate variables. During structuring, the optimization algorithms search for sub-functions that minimize logic equations. Both “Flattening” and “Structuring” operate on the logic level and are technology independent. The following step operates at the gate level, and is technology dependent:

3) Mapping - also known as “Technology Mapping”, is the phase in which the synthesis tool selects from the technology library (target library) components to implement the logic structure. The goal in this phase is to synthesize a gate-level implementation of a design that meets the **timing** and **area** constraints.

Three independent factors determine the ability of the synthesis tool to achieve an optimal result: The synthesis algorithms, the place and route (P&R) tool and the target cell library. Each element is constrained by the other two: During mapping, the synthesis algorithms have to map a given design into the cells provided by a particular library. The P&R tool has to route the resulting netlist of cells produced by the synthesis tool. In the currently available synthesis tools, neither the synthesis algorithms nor the place and route tools are capable of optimizing a particular design for **minimum power**. Instead, only speed and area are included in the objective functions and optimization constraints. In other words, cells from the target library are **selected** to meet the timing constraints with a minimal area implementation. Under these circumstances, the only way to ensure low power technology mapping is through the target cell libraries, which have to be especially designed for low power. Furthermore, the cells should be designed for an optimal utilization to take place during technology mapping.

3.3 Multiple Drive Cells

3.3.1 Drive Capability

The “drive capability” of a cell is the maximum capacitive load that can be charged/discharged per unit time. For a particular cell, this value is derived from the slope of the curve obtained for the rise/fall time as a function of different load capacitance. Since the amount of current that can be drawn at the output stage of a cell determines the rise and fall times, the “drive” in some cases is referred to as the “current drive capability” of the cell.

3.3.2 Cell Utilization During Technology Mapping

Providing library cells with a variety of drive strengths (for each cell in the library), has proven to be a useful method to increase the speed of synthesized designs [6]. When design for low power is the issue, multiple drive strengths might be important for an opposite reason: They can **slow down** the circuit speed at places where slower operation does not lead to an overall degradation of performance. For example, non critical paths.

Let us consider a specific case, where Synopsys has to select a “Nand2” gate during “technology mapping”. If the Nand2 cell is provided with various drive strengths, then the decision of which instance to select is based on the **timing** attributes (rise, fall and delays) and the “**area**” (cell area) attribute of the cell, in the Synopsys model (Appendix B). If all instances meet a specified timing constraint, then the Nand2 instance with the smallest “area” attribute is selected for mapping. In the following sections, it will be shown that cells with low current drive dissipate less power than those with higher drive. Therefore, it would be desirable in terms of low power that during technology mapping the synthesis tool selects cell instances with low drive whenever possible. This implies that the “drive capability” of a cell should be reflected in the “area” attribute since cost functions for power do not exist. Thus cells with high drive should have larger “area”

than cells with low “drive”, and vice versa. Section 4.5 provides further discussion on this issue.

3.3.3 Power and Delay Characteristics

The difference in the power dissipation of cell instances is caused by the switching and short circuit terms in equation 2.1. Two scenarios have to be considered in this context: The first is when a particular cell is driven by another, and power is dissipated due to the switching of the fanout gate. This component of power is a function of the gate capacitance (equation 2.1). As mentioned earlier, lowering this capacitance results in a direct reduction of the power.

Considering a simple inverter, the magnitude of the current drawn through the transistors, as well as the gate capacitance, are both functions of the gate widths W_p and W_n . Hence, the current drive capability and the input capacitance are closely related through these parameters. When combining these facts, it is apparent that toggling inverters with low drive capability decreases the power dissipation in comparison to inverters with large drive. This phenomenon can be generalized for all single stage cells, where the input port determines the current drive at the output.

The second scenario is when a particular cell is driving a fixed load: The non-ideal rise and fall times (ideal is zero) at the input, cause both the nmos and pmos transistors to be active at the same time, for a short period. In Figure 3.2, a simple inverter is driving a fixed load C_L , with a non ideal input waveform V_{in} . Based on the CMOS inverter’s DC transfer characteristic and operating regions (Table 3.1), between times t_1 and t_2 the pmos and nmos transistors switch between the “linear” and “saturated” regions. This may result in a short circuit current path from V_{dd} to Gnd, that increases the overall power dissipation.

Figure 3.2: Short Circuit Current for Non-Ideal Input Signal

Condition	P-Tran. Region	N-Tran. Region
$V_{tn} < V_{in} < V_{dd} / 2$	Linear	Saturated
$V_{in} = V_{dd} / 2$	Saturated	Saturated
$V_{dd} / 2 < V_{in} < V_{dd} - V_{tp} $	Saturated	Linear

Note: Parameters assumed in this table: $V_{tn} = -V_{tp}$, and $\beta_n = \beta_p$.

Table 3.1: Transistor's Region of Operation, Between t_1 and t_2 [29]

The magnitude of I_{sc} depends on the widths of the transistors (W_p and W_n), both in the “Linear” and “Saturated” regions. Variations of W_p and W_n (different drive strengths) cause changes in the short circuit power component (P_{sc}) when charging/discharging the fixed load. Even though P_{sc} is around 10% or less of the total power in a properly designed cell [29], this component may add up and become significant in large designs.

The following figures show the HSPICE simulation results of the total average power dissipation and delays of three inverters with different gate widths. In each case, the output load capacitance has been varied from 0.01pf to 0.2pf, in steps of 0.01pf.

The input rise/fall times in the simulations were 3ns since it was a “worst case” specification for the design and characterization of the library (section 4.3.2).

Figure 3.3: Average Power [wt] vs. Load Cap. [F] for Different Inverter Instances

Figure 3.3 shows a clear difference in the power dissipation of these inverters, with significantly larger values for the inverter with the largest transistor widths. Obviously, this experiment demonstrates an extreme case, because the worst case 3ns rise/fall time is quite high, and the gap between the curves is expected to be much smaller for input rise or fall times which are less than 3ns. However, these simulations demonstrate the power saving that can be achieved by using cell instances with low drive, rather than instances with high drive.

The following figure shows the simulation results of the delays, for the same inverters. The delays were measured from $V_{in}(50\%)$ to $V_{out}(50\%)$, and the graphs show the output low to high transition times.

Figure 3.4: Delay [S] vs. Load Cap. [F] - Inverter Cell Instances

There is a speed penalty associated with the use of low drive cells (equation 2.3). Cells realized by transistors with high W/L ratio can draw more current per unit time, thus charging a given load capacitance much faster than a cell with low current drive capability. Nevertheless, when comparing curves A and C, it can be seen that for relatively small capacitance (less than 40fF), the delay in curve A is only twice as much as in curve C, although the gate widths ratio of the two cells is almost four ($W_{pC} / W_{pA} = W_{nC} / W_{nA} = \sim 4$).

3.4 The “kcell” Library

The “kcell” standard cell library, designed at NT/BNR, has been used as a benchmark and reference for this thesis. It was specifically designed for low power purposes, and served as target library for several ASICs. It contains approximately 140 cells, including simple and complex logic gates, muxes, and flip-flops.

3.4.1 Technology

The cell library is targeted for NT's 0.8 micron BATMOS process, thus all corresponding design rules are followed. The cells are designed to operate with a nominal 3.3V supply voltage.

3.4.2 Logic Style

Conventional static CMOS is the logic style used for the implementation of the majority of cells in the library. Pass transistors are utilized in flip-flops, XORs and muxes, in order to increase speed.

3.4.3 Transistor Sizing

The main consideration for transistor sizing has been to create a balance between providing cells with small input capacitance, yet large output drive capability, to maintain performance. The other main factor was reliability driven: At the output stage of a cell, the minimum transistor width should allow two source and/or drain contacts, as shown in Figure 3.5. Based on these concerns, the BATMOS design rules define the minimum allowed gate width of an output stage to be 3.8 μm . The gate length is 0.8 μm for all the cells in the library. The transistors are sized according to the following cell classifications: Single Stage Cells and Multiple Stage Cells.

3.4.3.1 Single Stage Cells

For these cells, the input port directly gates the output stage, or in other words, the input capacitance and the output drive of the cell are a function of the size of the same transistors. Therefore, in order to create a uniform specification for the cells, the controlling factor in sizing the gates has been chosen to be the input capacitance: All the X1 drive cells have the same input capacitance, and likewise the X2 and X4 drives. For example,

Figure 3.5: Defining the Minimum Gate Width

the input port capacitance of knr3_2 (three input NOR with X2 drive designation) should be the same as the input capacitance of knd5_2 (five input NAND with X2 designation). Consequently, two different cells with the same drive designation may be substantially different in terms of speed and delays.

The input capacitance values of the different cell instances were determined after sizing the X1 inverter: The minimum width of the nmos transistor is $3.8 \mu\text{m}$, according to the previously described guidelines. Based on the choice of nmos transistor, the best performance (speed, power) of the corresponding inverter was obtained for a $6.8 \mu\text{m}$ pmos transistor. These values defined the X1 inverter's input capacitance to be 0.019 pF . The transistor widths and input capacitance of the other drive versions (of the inverter or any other cell) are integer multiples of the X1's values. Table 3.2 summarizes the input pin capacitance and the sum of transistor widths of the different drive strengths.

Drive	Cin [pF]	Wp+Wn [μm]
X1	0.02	10.6
X2	0.04	21.2
X4	0.08	42.4
X8	0.16	84.8

Note: Inverters and buffers are provided with additional drive strengths - X6 and X16, that are sized in the same manner.

Table 3.2: Fixed Input Capacitance and Transistor Widths

3.4.3.2 Multiple Stage Cells

Multiple stage cells have more flexibility in sizing the transistors, since the input and output stages are separate. These cells have an output stage which is identical to the output stage of a corresponding inverter of the same drive. For example, the X2 AND gate should have an output stage that matches the X2 inverter transistor sizes. At the input stage, these cells have an input capacitance less than or equal to the corresponding inverter of the same drive strength. The sizing of the internal stage transistors varies according to performance considerations.

3.4.4 Performance

The sizing rules which were defined for the “kcell” library, resulted in uniform input loads and consistent structures, yet caused variations in delays (for a certain drive strength), and inconsistent rise and fall times. In order to limit these differences, the cells had to meet the following condition:

$$\frac{|t_{PLH} - t_{PHL}|}{t_{PLH} + t_{PHL}} \leq 0.333 \quad (3.1)$$

This ensures that in the worst case, the largest delay will not be larger than 100% of the smallest delay. Cells that failed to meet this criterion were not included in the library.

3.5 Modifying the “kcell” Library

3.5.1 New Drive Strengths: Lower than X1

Since the **minimum allowed** transistor sizes at the input and output stages in the “kcell” library are different from the **minimum possible** sizes (as defined by the BAT-MOS design rules), extra power is dissipated when the transistors are being switched. As explained in 2.3.1.2, scaling down the transistor sizes translates into reduction of the gate capacitance, and consequently in the overall power of the circuit. Hence, two new drive strengths that are lower than X1 are introduced in the modified library which has been developed for this thesis: X0p75 and X0p5. These cell instances differ in their gate capacitance, as well as their speed and delays.

3.5.2 New Drive Strengths: Between X1 and X2

Analysis of several HDL designs mapped to the “kcell” library shows that the utilization of X1 drives is in the range of 70%, and that of X2 cells is approximately 20% (Chapter 6). To examine the possibility of further reducing the gate capacitance (contributed by the X2 cells), the following new drive strengths were created and placed in the modified library: X1p25 and X1p5. The idea is to enrich the selectivity during technology mapping, in the following cases:

1. When timing constraints are met, some of the X2 drive cells can be replaced by the new drives, which are slower, but dissipate less power.
2. The schematics of several synthesized designs show that buffers were randomly cascaded at the output of X1 cells. It implies that the buffers have been placed in order to

meet timing constraints (on a certain path). By providing cell instances with slightly larger driving capability (than X1), the number of such buffers can be reduced.

The same HSPICE simulations which were described in Section 3.3.3, were carried out for all the drive instances (including the new ones) of a two input NAND gate.

Note: The transistor sizes of the various drive instances are listed in Table 4.1.

Figure 3.6: Average Power [wt] vs. Load Cap. [F] - Nand2 Cell Instances

As seen in Fig. 3.6, the X0p5 and X0p75 drive versions dissipate less power than the X1 drive, and the curves of the X1p25 and X1p5 versions fit into the gap between the X1 and X2 drives, as expected. The gaps between the curves are rather small, and in practice may even be smaller if the rise/fall times of the input waveforms are less than 3ns. However, this experiment is a good indication for the feasibility of saving power by the utilization of the new drive strength versions.

Even though the delays drastically increase in the X0p5 and X0p75 versions, it is assumed that during technology mapping cell instances will be “intelligently” selected by the synthesis tool, based on the specific timing constraints.

3.6 Summary

In this chapter, the basics of the logic synthesis methodology were introduced, as well as the role of technology specific target libraries, and the way in which cell instances are selected during technology mapping.

The power and delay characteristics of cells with multiple drives were discussed. It has been shown that cell instances with low drive capability dissipate less power than those with high drive, therefore it is preferable to select those with the lower drive during technology mapping.

Finally, the “kcell” library was presented, and the ways in which it could be improved were discussed.

Chapter 4

The Implementation of the Multiple Drive

Library

4.1 Design Considerations

Since the utilization of the logic gates/cells is not uniform, and for practical reasons (long development process and maintenance), it has been decided to provide the new drive versions (X0p5, X0p75, X1p25, X1p5) only for those logic cells that are most often selected during synthesis. Analysis of several ASICs (mapped to the “kcell” library) identifies **twenty five** such cells, and only those are delivered with the new instances (Appendix A). Accordingly, 100 new cells form the “modified kcell” library which contains a total of 248 cells. In the rest of this text it will be referred to as “**kcell.p2**”, and the original “kcell” library will be referred to as “**kcell.p1**”.

Since the kcell.p2 library is comprised of cells from kcell.p1, it is important to design, test, and implement the new cells according to the same specifications used for the kcell.p1 library. Changing the design considerations and definitions would necessarily create bogus results. Therefore, based on the original definition: A particular “drive” instance is distinguished by its **constant** input capacitance. As a result, the **sum** of the corresponding pmos and nmos transistors is constant as well. Furthermore, the new cells have to be

sized in a linear fashion with respect to the other drive instances, as shown in the following table:

Drive	Cin [pF]	W _p +W _n [um]
X0p5	0.01	5.3
X0p75	0.015	8.0
X1	0.02	10.6
X1p25	0.035	13.3
X1p5	0.03	15.9
X2	0.04	21.2

Table 4.1: Input Capacitance and Transistor Widths

The rest of the drives (X4, X8 etc.) remain alike those in the kcell.p1 library. All cells are realized with minimum length transistors ($L = 0.8 \mu\text{m}$), in order to maintain speed.

4.2 Transistor Sizing

Since the sum of transistor widths is fixed, the only variable which has to be determined for the design of a single stage cell, is the ratio between the nmos and pmos transistors. The general guideline is to keep the transistor sizes as close as possible to the integer multiples derived from the transistors of the X1 inverter. These requirements leave a very small margin of flexibility when sizing the transistors. However, the predominant concern in the limited margin is to achieve minimum delays with minimal differences between rise and fall times. Figure 4.1 shows the experimental procedure which has been used to determine the transistor values of the single stage cells.

Figure 4.1: Experimental Procedure to Determine Best Rise/Fall Times

For a given drive instance (device under test - D.U.T), the pmos and nmos transistor widths are varied around the transistor values of the corresponding drive inverter (derived from the X1 inverter), such that optimal performance (with respect to delays and timing) is achieved. The purpose of the left-most inverter is to shape the input signal, and the two right-most inverters serve as a fixed load capacitance at the output of the D.U.T.

4.3 Layout Format

4.3.1 Cell Dimensions and Topology

Figure 4.2 summarizes the general layout structure, used for the standard cell library. Since the “*Cadence Gate Ensemble*” is the place and route tool, all the standard cells have to align to a fixed routing grid with respect to I/O port placement, and cell boundaries. The cells are placed in a tile arrangement (without channels), with overlapping supply rails. The routing is done by a grided maze router (over the cell routing). This constrains the area under the supply rails, as well as the cell width and the I/O port locations that have to be placed on a grid. Only Metal1 is allowed for routing within the cells, and Metal2 is reserved for intercell routing (by the P&R tool).

Note: The x and y grids are not shown in this figure.

Figure 4.2: Layout Format for Standard Cell Library

The transistors are aligned horizontally, with their width parallel to the vertical axis. The total cell height (Y1) is fixed for all cells. The cell width (X1) can be varied according to the logic implementation of the individual cells. The VDD supply rail should always be on the top and VSS on the bottom. The bounding box of a cell is formed by the top of the VDD rail, the bottom of the VSS rail, and the sides of both rails that are on a grid. The only layer which can extend beyond the bounding box is the N-Well layer.

4.3.2 Layer Constraints

In order to allow flexibility in the cell design, there are maximum and minimum sizes for the N-Well (Y2 in Figure 3.6 is the minimum size). Since the cells are tiled horizontally, the cells are designed in such way that N -Well incisions will not be formed by the projection of a maximum N-Well from a neighbouring cell. The sides of the N-Well must always overlap the cell boundary (X2 in Figure 3.6).

The N-Device and P-Device diffusions must be at least X3 microns inside the cell boundary (minimum spacing between active areas of the same type). The placement of the N-Dev. should take into account a possible maximum N-Well from the adjacent cell.

All the cell I/O ports have the necessary layers (MET1, vias, labels etc.) required by the design rules and the P&R tool. The access directions are set on the I/O pins (although not required by *Gate Ensemble*), with the supply rails having LEFT and RIGHT access only, and the I/O pins having TOP and BOTTOM access only.

In order to avoid design rule violations between two adjacent cells, the polysilicon and MET1 layers have to keep a specified distance from the cell's bounding box.

4.4 Library Development Phases

The task of generating and maintaining a complete cell library, including the layouts, symbols, models, and characterization, is a major effort that requires the extensive use of automated software. Figure 4.3 shows the development phases of the kcell.p2 library, including the software tools which were used. Detailed explanation about the library models and development process will follow later on.

Figure 4.3: Library Development Process

4.4.1 Physical Layouts

The first step in developing a new library, is to create all the physical layouts of the cells, based on the transistor sizes which had been previously determined by simulations. For this thesis, 100 layouts were generated by using *Analog Artist*. The layouts of the new drives are based on the corresponding X1 cells from kcell.p1. In other words, the existing X1 layouts were modified to fit the transistor sizes of each one of the new drives. As a result, the “real” **cell area** of a particular cell in kcell.p2, is identical for all the new drive instances. Therefore, the “area” attribute in the Synopsys models needed to be modified, as explained in Section 4.5.

The next step after generating the layouts, is to create a “post-layout” transistor level netlist for each cell. For this work, HSPICE format netlists were extracted directly from the physical layout (using *Analog Artist*).

4.4.2 Cell Characterization

Characterization is the process in which the performance of the cells is evaluated, and the information is provided in a text format to the synthesis tool (Synopsys) and the

logic simulator (Verilog), to enable accurate timing calculations for these tools. This information includes data such as pin-to-pin delays, rise/fall times, drive capability and input pin capacitance of all the cells in the library. The information is obtained by running HSPICE simulations on the post-layout (extracted) version of each cell. The simulations are carried out for BEST, TYPICAL and WORST case technology parameters.

The characterization process has been carried out for the following data points: For a rising output, the propagation delay t_{PLH} is the time interval between the input reaching 50% of its final value and the output signal rising to 50% of its final value. For a falling output, the propagation delay t_{PHL} is the time interval between the input reaching 50% of its final value and the output signal falling to 50% of its final value. The rise time t_r is the time interval between the output rising from 10% to 90% of its final value, and the fall time t_f is the time interval between the output falling from 90% to 10% of its final value. The “final value” is considered to be the rail potential, since it is a CMOS library. The maximum drive capability of a cell is defined as the maximum capacitive load at the output that can be charged/discharged in 3ns (worst case rise/fall time of 3ns).

Figure 4.4: Measurement Points for Timing Characterization

An important issue to understand prior to setting up the characterization platform, is the delay model used by the synthesis tool for timing calculations. Knowing these models and the required parameters is essential for properly setting up the simulation deck, and for obtaining valid results. Three different delay models are supported by Synopsys, and in this work, the “CMOS Standard Delay Equations” [30] are used. It is a linear model which performs pin-to-pin delay calculations during synthesis.

For large size libraries, the characterization process is often automated, and for this thesis, *ACCELL*, a BNR/NT proprietary software was used. After the HSPICE input deck is set up, it runs the simulation, and then extracts the necessary information into an ASCII text file in a special format. This file is then provided as input to another proprietary software that creates the Synopsys models (Section 4.4.3). The process is repeated for all the cells in the library.

4.4.3 Synopsys Library

In order to use the technology specific libraries for mapping synthesized designs, a proper representation should be provided to the “*Synopsys Library Compiler*”, which compiles ASCII text description into an internal database format:

Figure 4.5: The Role of Library Compiler

Two types of libraries need to be created prior to compilation: *Technology Library* and *Symbol Library*.

4.4.3.1 Technology Library

The technology library is a text file that contains the characteristics and functionality of each cell in the library. It contains four different types of information:

- 1. Structural Information** - Describes each cell's connectivity to the outside world, including bus and pin description.
- 2. Functional Information** - Provides the logical function of every output pin (as a function of the inputs).
- 3. Timing Information** - Provides the pin-to-pin timing relationships and the delay calculations. Setup and hold times must be provided for sequential cells. This data is obtained from the characterization results.
- 4. Environmental Information** - Contains data such as manufacturing process, operating temperature, supply voltage variations, wire capacitance and resistance, and scaling factors for variations in the process.

4.4.3.2 Symbol Library

This library contains information on the graphic symbols that represent each cell, the page borders and off-sheet connectors. It enables *Design Analyzer* to draw schematics of designs on the computer screen.

The following figure shows the typical structure of the technology library:

Figure 4.6: Synopsys Technology Library Structure

The above shown text structure was created by using another NT/BNR proprietary software, which translates the ASCII text file created by *ACCELL*, and additional technology related data into the proper format (Appendix B).

4.4.4 Verilog Library

The Verilog logic simulator was chosen for the simulations of the synthesized designs. The utilization of this tool requires **all** the cells in the library to be represented by a special Verilog model. The model is an ASCII text file, which describes the logic func-

tion of the cell, its connectivity to the outside world, detailed description of the pin-to-pin delays, rise/fall times, and input pin capacitance. Each model is placed in a separate file, which contains additional information such as scaling factors and compiler directives. The proper format for the models can be directly extracted from *ACCELL*.

4.4.5 EPIC Library

“PowerMill” has been the tool of choice to carry out the power simulations. As will be discussed in the following chapter, the simulations are carried out at the transistor level, thus proper netlist format is needed for the representation of the cells. The “*spice2e*” utility program (part of the PowerMill package) translates an HSPICE netlist to the equivalent EPIC format [31].

4.5 Additional Library Versions

One way to add a measure for the power dissipation can be done by modifying the area attributes of the cells, and using them as an artificial* cost function for the power dissipation of each cell. It directly affects the selection of drive instances during technology mapping (Chapter 6). The purpose is to “encourage” the synthesis tool to select the cell instances with the lowest possible drive. For a given cell with the new drive versions in the kcell.p2 library, the “real” cell area of all instances was identical to the area of the corresponding X1 instance. In order to distinguish between the different drives (in terms of power), the “area” attributes have been scaled by adding a “fudge” factor of X square microns to the “real” cell area:

$$\text{Modified_Cell_Area} = \text{Real_Cell_Area} + X \quad (4.1)$$

All the values are in square microns. The following table shows the “X” factor which is added to each one of the corresponding drives:

* Optimization for minimum power is not available yet in synthesis tools.

Drive Instance	X0P5	X0P75	X1	X1P25	X1P5	X2	X4
X Factor [μm^2]	2	3	4	5	6	8	16

Table 4.2: “Area” Attribute Modification Factors - kcell.p2 Library

After extensive synthesis and simulations using the kcell.p1 and kcell.p2 libraries (Chapter 6), a few changes were implemented in the kcell.p2 library, in order to investigate the possibility of improving the cell utilization during technology mapping, and to further improve the power dissipation of the synthesized designs. These changes did not necessitate carrying out all the steps described in the previous section, thus, the new libraries were only supplemental versions of the kcell.p2 library.

4.5.1 The “kcell.p3” and “kcell.p4” Versions

The main purpose for creating the **kcell.p3** version is to investigate the possibility of increasing the utilization of the X1p25 and X1p5 drive instances by decreasing the utilization of X2 drives. The only change in this version in comparison to kcell.p2, is the “area” attribute in the Synopsys models: Instead of adding an incremental factor of “X” square microns, the “area” attributes are normalized with respect to the X2 drive instances, according to equation 4.2.

$$\text{Modified_Cell_Area} = \text{Real_Cell_Area} * [\text{Drive_Strength} / 2] \quad (4.2)$$

The “Drive_Strength” term is the **numerical value** of the corresponding drive. For example, if the “real” cell area of an X0P5 inverter is $100 \mu\text{m}^2$, then the “modified” area attribute would only be $25 \mu\text{m}^2$, whereas for an X2 inverter with a “real” cell area of $200 \mu\text{m}^2$, the “modified” area would remain the same. This scaling method creates a situation where the difference between the area attributes of the X2 drive instances and the

smaller ones (X1P5 and below) is much larger than in the previous version (kcell.p2), so the cost function of the X2 drive cells appears to be more “expensive” during synthesis.

The “kcell.p4” library version is exactly the same as the “kcell.p2”, except for the two additional drive instances that were added to the “kdf” cell (D-type flip-flop), originally provided with X2 drive. The new instances are X1 and X1P5.

Chapter 5

Experimental Procedure

5.1 Introduction

Section 3.2 presented the general concepts of the synthesis methodology, including the role of technology specific target libraries. The idea is to use those concepts in order to compare the performance of various libraries (with respect to power). Several HDLs, originally designed and implemented for telecommunications applications, are used as benchmarks for the technology mapping. Each HDL is mapped to the various libraries, and simulations for power consumption are carried out on the resulting gate level implementations (netlists). For the simulation purposes, the “PowerMill” simulator and the “Vertue” interface have been integrated into the existing BATMOS digital design flow. Figure 5.1 shows the entire system, including the required library models.

5.2 Synthesis and Power Simulation Methodology

The following figure represents the experimental infrastructure that was put in place for the synthesis and simulations:

Figure 5.1: Synthesis and Power Simulation Methodology

As seen in Figure 5.1, several CAD tools are involved in the realization of the proposed methodology for simulating the power dissipation. The system consists of two main parts: Synthesis and Power Simulation.

5.2.1 Synthesis Environment

Figure 5.1 presents the specific components of the Synopsys synthesis tool that are involved in the process of converting an HDL design into a gate level netlist. Since the benchmark designs are written in Verilog HDL, the *HDL Compiler* reads and translates the design to the internal data-base representation. The cell libraries are compiled within *Library Compiler*, thus creating a data-base for each one. The synthesis is carried out in *Design Compiler*, and the resulting implementations are saved in Verilog format, to allow simulations with the Verilog logic simulator.

5.2.2 Power Simulation Environment

The core of this environment and the most essential component is the PowerMill simulator, which accurately simulates the power consumption of a given design. Since PowerMill is a transistor level simulator, and the synthesis result is a gate level netlist, the “Vertue” software has been used. This software is an interface between Verilog and PowerMill. Thus, PowerMill becomes transparent to the Verilog user, and the original Verilog test bench and models (Figures 3.1 and 5.1) can be applied.

In order to integrate Vertue into the simulations, the Verilog netlists are partitioned into a Vertue data-base, called the “Verilog & Vertue co-simulation Environment” (Figure 5.1). The stimulus vectors can then be applied, and based on the switching activity and simulation events, PowerMill provides the power information.

5.3 Preparing PowerMill for Simulation

5.3.1 PowerMill's Features and Capabilities

PowerMill is currently the only available simulator which can accurately **simulate** the power consumption of designs containing more than 50,000 transistors. Transistor level simulators like HSPICE can simulate very small circuits, whereas gate level tools only perform power **estimation** based on probabilistic computations or monitoring the switching activity of nodes.

Being a transistor level tool, PowerMill is capable of handling the full spectrum of CMOS digital circuits. It employs a piecewise linear transistor model which captures the transistor characteristics in look-up tables [28, 31]. The look-up tables are the main reason for the superior speed and circuit sizes which can be simulated, compared to SPICE-like tools. In contrast to gate level simulators, events are determined in terms of small voltage changes, rather than logic transitions. Thus, non-digital behavior (such as glitches) can be accurately captured. The overall accuracy of the simulations is within 10% of HSPICE, provided that a proper technology file is generated.

5.3.2 Epic/BATMOS Technology File

The Epic technology ("tech") file is the engine of the simulator, therefore its accuracy is crucial for obtaining reliable simulation results. It contains technology specific parameters, as well as "look up tables" of the drain-source current (I_{DS}) versus V_{GS} , for different size nmos and pmos transistors. Technology specific information needs to be extracted into an Epic "control" file, which is then applied as input to **gentech** (an Epic utility program), which creates the "tech" file. For this thesis, the "control" file contains all the necessary data from the HSPICE models of the BATMOS technology, for "Typical" process parameters.

An internal mechanism exists for checking the accuracy of the “tech” file. HSPICE and PowerMill simulations are carried out on pre-defined circuits, and the results are compared. The reports obtained for the BATMOS “tech” file indicated very good accuracy: Less than 10% difference between the results of both simulators. To further examine the “tech” file, additional simulations were carried out on different test circuits, using both HSPICE and PowerMill. The results are listed in the following table:

Circuit	No. of Transistors	PowerMill $I_{avV_{dd}}$ [μA]	HSPICE $I_{avV_{dd}}$ [μA]	Accuracy [PowerMill] [%]
One Inverter	2	16	18.3	14
Chain of 5 Inverters	10	149	158	6
Chain of 5 Nand Gates	20	340	373	9.7
Clock Buffer	40	2685	3004	11

Table 5.1: PowerMill vs. HSPICE Simulation Results

As seen in Table 5.1, the simulation results from both tools are very close. At this point, it means that the generated “tech” file for the BATMOS process is accurate, and can be used for the PowerMill simulations.

5.3.3 Translating the Synthesis Result to an “Epic” Netlist

In order to carry out PowerMill simulations, the circuit has to be represented in a proper Epic format. The synthesis result, in this case a gate level Verilog netlist, should be translated to this format. The conversion is carried out by the **vlog2e** utility program. The Epic netlisting format supports hierarchical structure, and is based on “sub-circuit” defini-

tions. The top level modules are propagated down along the hierarchy by “sub-circuit” calls. The transistor level netlist which has to be created for each cell (section 4.4.5) is at the lowest level.

5.4 Statistical Wire Load Models

The wire load models are part of the environmental description in the Synopsys technology library. These models provide information on the capacitance and resistance of interconnect wires. For the initial synthesis of a design, where no layout back annotation of the parasitic capacitance is available, the wire load models have a significant impact on the synthesis results. In this thesis, layouts were not available, thus the information regarding interconnect capacitance was based on **statistical** wire load models, derived from the layouts of several chips fabricated in NT’s BATMOS process. These models were extracted from three different size logic blocks, thereby creating a measure for the SMALL, MEDIUM and LARGE wire load models. The difference is in the capacitance values associated with the interconnect wire lengths (Appendix B).

The wire load models are specified as optimization parameters during synthesis. As shown in the next chapter, the choice of wire load model affects the synthesis results, especially the number of cells. Hence, different power simulation results are obtained, depending on the wire loads. Since none of the benchmarks is in the category of “LARGE” design (the largest is ~7800 cells), applying this wire load model creates an overly aggressive scenario where the interconnects have a very significant effect on the delays and power. This is useful when trying to estimate the merits of multiple drive libraries in more advanced sub-micron technologies (i.e. 0.5 μm , 0.35 μm , etc.).

5.5 Benchmark Designs

Three HDL benchmarks were synthesized and simulated in order to find out whether the proposed multiple drive libraries result in better implementation and reduced power, as compared to the kcell.p1 library. Two benchmarks were designed at NT/BNR for telecom applications, and the third (Data-Path) appears in Appendix C.

5.5.1 Data-Path

The first benchmark is a very basic Data-Path unit (Figure 5.2), which has three 4-bit words at the input: It checks whether the sum of the first two (A, B) is greater, equal or smaller than the third (C). The Synthesis result of the HDL contains approximately 60-70 cells, depending on the target library.

Figure 5.2: Basic Data Path Unit

5.5.2 Data-Path Control Unit (DCU) - B62

The B62 is a multi-channel signal processor chip which resides on a peripheral interface card, designed at BNR/NT. It provides typical signal processing to 32 lines simultaneously, using DSP technology. The DSP module of the chip includes three major parts: data-path, data-path control unit (DCU), and memories. Only the HDL of the DCU

was used for the experiments, and its synthesis resulted in 2500-3500 cells (Depending on the constraints). The main functionality of this unit is to decode micro and macro instructions of the DSP, and to provide the appropriate control signals to the data-path. In addition, it formulates the memory addresses.

5.5.3 Programmable Line Card Controller - A34

The A34 Line Card Controller (LCC) device is a high speed microprocessor component that includes on-chip program and data memories as well as interfaces to other chips. It has been designed for application in one of BNR/NT's line cards. The main functional block of the A34 is the "processor" block, which can execute common arithmetic and logical operations at a very high speed. A large number of function circuits are closely coupled with this block, and all the "on-chip" interfaces are accessible by the processor.

Chapter 6

Experimental Results

6.1 Simulation and Synthesis Results: Data-Path

The HDL of the Data-Path was synthesized with various timing constraints, clock frequencies and wire load models, so that a different mapping (implementation) would take place with each set of constraints. This way, a large variety of simulations was carried out for a particular target library, and the comparison of power consumption could be performed over a broad range of results.

6.1.1 Wire Load Model - SMALL

The SMALL wire load model was globally set on all blocks, thus modeling a moderate effect of the interconnect wires. The results are summarized in Table 6.1, and the following explains the terminology used in the table: Design “DP_A.S” is the synthesis implementation result when the clock frequency is set to 500 MHz ($T_{ck}=2ns$). For “DP_B.S” $F_{ck}=250MHz$, for “DP_C.S” $F_{ck}=125MHz$ and for “DP_D.S” $F_{ck}=62.5MHz$. The “.S” notation specifies the wire load model (SMALL).

The power simulation results are in the form of total average current, consumed from the source ($I_{av_{V_{dd}}}$). The exact value of the power consumption in watts can be calcu-

lated by multiplying the current by the voltage source value (which is 3.3V).

Library	KCELL.P1	KCELL.P2	Power Saving: P2 vs P1[%]
Design	I _{avV_{dd}} [mA]	I _{avV_{dd}} [mA]	
DP_A.S	3.00	2.421	-23
DP_B.S	1.82	1.41	-29
DP_C.S	0.925	0.709	-30
DP_D.S	0.471	0.356	-32

**Table 6.1: Power Consumption of the Various Data-Path Implementations
(Wire Load Model: SMALL)**

As shown in Table 6.1, it is quite evident that significant power saving can be achieved by using the modified multiple drive library (kcell.p2). The power reduction is consistent throughout all synthesis scenarios (clock frequencies and timing constraints), and is in the range of 23%-32%. Most of this extremely promising decrease in power, can be attributed to the fact that the SMALL wire load model was used, and the timing constraints (including the most aggressive) could be easily met even when selecting cells with low drive. Table 6.2 provides more information on this issue, and it shows the cell utilization for implementation DP_A.S: The total number of cells required to implement the functions was almost the same, 62 vs. 67, but the vast majority of drive instances when using kcell.p2 were the X0p5 instances (63%), as opposed to X1 drives when using kcell.p1. Replacing the X1 drives with X0p5s, results in substantial reduction of the overall capacitance in the design, thus significantly less power is dissipated.

Library	KCELL.P1	KCELLP2
Total no. of Cells	67	62
Drive		
X0P5[%]		63
X0P75 [%]		6
X1 [%]	80	2
X1P25 [%]		2
X1P5 [%]		
X2 [%]	20 (*20)	27 (*22)
Others [%]		

Note: The (* “number”) notation represents the percentage of sequential cells with this drive instance.

Table 6.2: The Drive Utilization of DP_A.S

All the X2 drive cells selected for mapping when the target library is kcell.p1, are comprised of sequential cells, and similar results are noticed for the kcell.p2 library. Since the majority of these cells are exclusively provided with X2 drive instances (in both libraries), additional power reduction can be expected if the sequential cells are offered with multiple drives as well.

6.1.2 Wire Load Model - LARGE

The only parameter that has been changed in this case is the wire load model: DP_A.L and DP_B.L (Table 6.3) were set up for synthesis with the same timing constraints as DP_A.S and DP_B.S respectively, except now with the LARGE wire load. In addition, the same stimulus vectors were applied during simulation. Four synthesis were carried out using this model. The simulation results of the total average current are presented in Table 6.3. Although the total current in both implementations slightly increases (compared to Table 6.1), the overall power reduction by using kcell.p2 is almost the same as before - between 20% to 32%.

Library	KCELL.P1	KCELL.P2	Power Saving: P2 vs P1[%]
Design	Iav _{Vdd} [mA]	Iav _{Vdd} [mA]	
DP_A.L	3.326	2.75	-20
DP_B.L	2.13	1.606	-32

**Table 6.3: Power Consumption of the Various Data-Path Implementations
(Wire Load Model: LARGE)**

Since the functionality of the Data-Path is so limited, synthesizing the HDL with either the SMALL or LARGE wire load models resulted in almost the same implementation (60 - 70 cells). However, it will be shown that for larger designs the wire load models have a more significant effect on the synthesis implementation, power, and amount of power saving.

Table 6.4 shows the utilization of drive instances in implementation DP_A.L. The most prominent difference when comparing to Table 6.2, is the percentage of X0p5 drive cells: For DP_A.L, only 37% of the selected cells were X0p5 instances, compared to 63% for DP_A.S. A significant increase in X0p75 and X2 drives can be observed as well.

Library	KCELL.P1	KCELLP2
Total no. of Cells	71	65
Drive		
X0P5[%]		37
X0P75 [%]		21
X1 [%]	70	8
X1P25 [%]		3
X1P5 [%]		
X2 [%]	30 (*20)	31(*21)
Others [%]		

Note: The (* “number”) notation represents the percentage of sequential cells with this drive instance.

Table 6.4: The Drive Utilization of DP_A.L

6.2 Simulation and Synthesis Results: DCU (B62)

Since the B62 has not been manufactured (by NT), both the synthesis timing constraints and the functional test bench were not available for this research. Hence, different optimization scenarios were carried out by applying different clock frequencies, and accordingly, different timing constraints. This resulted in a different technology mapping each time the set-up had been changed. As for the previous benchmark, the DCU was synthesized with a specific optimization scenario several times, each time mapped to a different target library. Both the “MEDIUM” and “LARGE” wire load models were used.

A Verilog test bench was created for the simulations. It had to be slightly modified each time the synthesis set-up was changed, to match the timing specifications. The operating frequency of some inputs was known, and the appropriate waveform could be applied. Random patterns were generated for inputs with unavailable specifications. In order to maintain accurate comparison between the libraries, the test bench monitored a

wide range of internal and external I/Os, to confirm that during a specific time frame, the same logic state occurs at given nodes (for all compared libraries).

6.2.1 Wire Load Model - MEDIUM

The first set of synthesis and simulations were carried out using the MEDIUM wire load model. Table 6.5 is the summary of the PowerMill simulation results, and contains the reports of the total average, capacitive and leakage currents.

Design		KCELL.P1 Average Currents [mA]	KCELL.P2 Average Currents [mA]	Power Saving (P2 vs. P1)
DCU_A.M Tck=13.8ns	Iav-total Iav-cap Iav-Leakage	3.67 2.54 1.12 (30%)	3.38 2.38 0.99 (29%)	-8% -6%
DCU_B.M Tck=55ns	Iav-total Iav-cap Iav-Leakage	2.131 1.38 0.748 (35%)	1.893 1.269 0.624 (33%)	-12% -9%
DCU_C.M Tck=80ns	Iav-total Iav-cap Iav-Leakage	1.558 0.963 0.595 (38%)	1.346 0.872 0.474 (35%)	-15% -10%
DCU_D.M Tck=160ns	Iav-total Iav-cap Iav-Leakage	0.892 0.48 0.411 (46%)	0.747 0.442 0.305 (40%)	-19% -8%
DCU_E.M Tck=240ns	Iav-total Iav-cap Iav-Leakage	0.673 0.32 0.353 (52%)	0.541 0.292 0.248 (46%)	-24% -9%

Note: The percentage of the leakage currents (of the total average) are shown in brackets.

**Table 6.5: Power Consumption of the Various DCU Implementations
(Wire Load Model: MEDIUM)**

In Table 6.5, DCU_A.M is the resulting implementation when the clock frequency is ~72MHz (Tck=13.75ns), and the timing constraints are set to 10ns. DCU_B.M is the synthesis result when the clock frequency is set to 18MHz (Tck=55ns), and the output timing constraints are set to 50ns. For DCU_C.M, the clock is 12.5 MHz (Tck=80ns) and the timing constraints are 70ns. For DCU_D.M, the clock is 6.25MHz (Tck=160ns) with 140ns timing constraints, and for DCU_E.M, the clock frequency is 4.166MHz (Tck=240ns) with 200ns timing constraints. The “.M” notation represents the “MEDIUM” wire load model.

The analysis of the total average current ($I_{av-total}$) obtained for DCU_A.M through DCU_C.M, indicates that a power reduction of 6%-15% is taking place when kcell.p2 is the target library. The simulation results of DCU_D.M and DCU_E.M, present a larger power saving (19%-24%), mainly caused by the relatively high “leakage” currents. This portion of the current is modeled by PowerMill as “leakage” due to the random input vectors and the slower clock frequencies that cause some of the nodes to be at “undefined” (X) or “high impedance” (Z) states. The “capacitive” (switching) portion of the current illustrates a power saving of 6%-10%, thus supporting the results obtained for DCU_A.M through DCU_C.M to be more realistic.

The power saving (when using kcell.p2) can be primarily attributed to the selection of X0p5 and X0p75 cell instances during synthesis (Table 6.6), which replaced the majority of X1 instances. Hence, the reduced gate capacitance in the final implementations prompted the power reduction. The X0p5 and X0p75 cells have lower drive capability than the X1 cells, and the outcome is a moderate increase in the total number of cells. Also noticed in Table 6.6, is the low utilization of X1p25 and X1p5 drive instances. It is caused by the fact that Synopsys already had three lower drive levels before it needed to select the X1p25 or X1p5 instances. As for the previous benchmark (Data-Path), it is apparent that the power could be further reduced by providing the sequential cells with additional drive instances.

Library	KCELL.P1	KCELL.P2	KCELL.P1	KCELL.P2
Design	DCU_A.M Tck=13.8ns	DCU_A.M Tck=13.8ns	DCU_B.M Tck=55ns	DCU_B.M Tck=55ns
Total No. of Cells	2691	2824	2595	2774
Drive				
X0P5 [%]		35		47
X0P75 [%]		8		8
X1 [%]	57	11	62	8
X1P25 [%]		4		3
X1P5 [%]		4		1
X2 [%]	39 (*19)	35 (*18)	35 (*20)	32 (*18)
Others [%]	4	3	3	1

Note: The (* “number”) notation represents the percentage of sequential cells with this drive instance.

Table 6.6: The Drive Utilization of DCU_A.M and DCU_B.M

6.2.2 Wire Load Model - LARGE

The second set of synthesis and simulations of the DCU were carried out using the LARGE wire load model. The other optimization parameters remain the same as they have been set for the MEDIUM wire load, including the test bench. Table 6.7 is the summary of the results. The “.L” notation represents the wire load model (LARGE).

The total average current for the first two implementations is almost identical, when using either the kcell.p1 or kcell.p2 libraries. An increase in $I_{av-total}$ occurs in the last two cases when mapping the HDL to the kcell.p2 library. As for the MEDIUM wire load model, the simulation results of the last two implementations are dominated by the high percentage of “leakage” current. Nevertheless, throughout all the simulations, the switching portion of the current indicates a reduction of 4%-5% when mapping the design

to kcell.p2 rather than kcell.p1. It is reasonable to assume that having the “real” test bench and optimization constraints would significantly reduce the amount of leakage currents, allowing the “capacitive current” to dominate the results.

Design		KCELL.P1 Average Currents [mA]	KCELL.P2 Average Currents [mA]	Power Saving (P2 vs. P1)
DCU_A.L Tck=13.8ns	Iav-total Iav-cap Iav-Leakage	3.549 2.556 0.992 (27%)	3.472 2.422 1.049 (30%)	-2% -5%
DCU_B.L Tck=55ns	Iav-total Iav-cap Iav-Leakage	1.998 1.397 0.6 (30%)	2.001 1.317 0.684 (34%)	~ Equal -6%
DCU_C.L Tck=80ns	Iav-total Iav-cap Iav-Leakage	1.419 0.965 0.453 (31%)	1.512 0.923 0.589 (38%)	+6% -4%
DCU_D.L Tck=160ns	Iav-total Iav-cap Iav-Leakage	0.756 0.487 0.269 (35%)	0.825 0.467 0.357 (43%)	+8% -4%

Note: The percentage of the leakage currents (of the total average) are shown in brackets.

**Table 6.7: Power Consumption of the Various DCU Implementations
(Wire Load Model: LARGE)**

Table 6.8 shows the selection of drive instances for DCU_A.L and DCU_B.L. It illustrates the effect of the wire load capacitance on the selection of cells: The utilization of X0p5 drive instances decreases by 12%-18% when the synthesis is carried out with the LARGE wire load model (Table 6.8), rather than the MEDIUM (Table 6.6). This fact, and the increase in the utilization of X0p75, X2, and higher drive instances are among the main reasons for limiting the power saving (Table 6.7).

Library	KCELL.P1	KCELL.P2	KCELL.P1	KCELL.P2
Design	DCU_A.L Tck=13.8ns	DCU_A.L Tck=13.8ns	DCU_B.L Tck=55ns	DCU_B.L Tck=55ns
Total No. of Cells	2744	2910	2654	2762
Drive				
X0P5 [%]		23		29
X0P75 [%]		14		17
X1 [%]	52	13	58	10
X1P25 [%]		4		5
X1P5 [%]		3		2
X2 [%]	42 (*19)	38 (*18)	36 (*20)	35 (*19)
Others [%]	6	5	6	2

Note: The (* “number”) notation represents the percentage of sequential cells with this drive instance.

Table 6.8: The Drive Utilization of DCU_A.L and DCU_B.L

6.3 Simulation and Synthesis Results: A34

The complete synthesis and simulation environments of the A34 were available for this research. Its HDL description was synthesized using both the MEDIUM and LARGE wire load models, and the resulting netlists are in the range of 7300-7800 cells (Table 6.10). The well defined simulation environment provided a good opportunity to make use of PowerMill’s capability to obtain power information of specific sub-designs.

The results are summarized in Table 6.9 (for both wire load models). The “.M” and “.L” notations represent the synthesis results obtained for the MEDIUM and LARGE wire load models respectively. “A34” is the entire synthesized design. The following blocks (sub-designs) were simulated:

1. “Processor”: is the main functional block of the A34.

2. “A37if”: is the block which provides interface to another chip (A37).
3. “Alink0”: synchronizes data transferred between a few other blocks.

Design	KCELL.P1 $I_{av-total}$ [mA]	KCELL.P2 $I_{av-total}$ [mA]	Power Saving (P2 vs. P1)
A34.M	15.544	13.44	-15%
Processor.M	4.51	4.00	-12%
A37if.M	3.472	3.285	-5%
Alink.M	2.06	1.676	-22%
A34.L	15.68	13.54	-15%

Table 6.9: Power Consumption of the A34

Mapping the A34 to kcell.p2 rather than kcell.p1 results in a significant reduction of 15% in the total average current, for both wire load models. As for the sub-designs, the amount of power saving varies, and depends on the cells in each block: The majority of the “A37if” is comprised of sequential cells, so the total current reduction is only 5% (multiple drive instances are not available for most of these cells). A reduction of 22% occurred for the “Alink” block, which is predominantly comprised of logic cells. The “Processor” block contains both sequential and logic cells, and a 12% decrease of the current took place.

Table 6.10 is the drive utilization analysis of the A34, for both wire load models. The high percentage of X0p5 cells, implies that timing constraints could be met even when using these low drive instances, and it is the main reason for the current reduction (Table 6.9).

Wire Load	MEDIUM	MEDIUM	LARGE	LARGE
Library	KCELL.P1	KCELL.P2	KCELL.P1	KCELL.P2
Total No. of Cells	7568	7355	7802	7610
Drive				
X0P5 [%]		55		43
X0P75 [%]		8		13
X1 [%]	76	12	70	12
X1P25 [%]		3		6
X1P5 [%]		1		1
X2 [%]	20 (*10)	20 (*11)	25 (*11)	22 (*11)
Others [%]	4	1	5	3

Note: The (* “number”) notation represents the percentage of sequential cells with this drive instance.

Table 6.10: Drive Utilization -A34 (MEDIUM and LARGE wire load models)

6.4 The Synthesis Results of the “kcell.p3” and “kcell.p4” Library Versions

6.4.1 Mapping to “kcell.p3”

As it can be seen in the previously shown drive utilization tables, the X1p25 and especially the X1p5 drive instances were the most rarely selected cells. The main purpose in creating the kcell.p3 library version (section 4.5) was to increase the utilization of these drives by reducing the selection of logic cells having X2 drive. All three benchmarks were mapped to the kcell.p3 version, and the utilization analysis show similar results: The number of X2 drive cells was reduced by 8%-10%. However, it did not result in additional X1p5 or X1p25 instances, and an increase in the X0p5 drive cells took place instead. This trend can be clearly seen in Table 6.11 (column “KCELL.P3”), which shows the utilization of cells when mapping the A34 to the various target libraries. Only 4% of the logic

cells remain with X2 drive, as opposed to 11% when kcell.p2 is the target library. Furthermore, an increase of the total number of cells is now required to keep performance (timing) due to the massive use of X0p5 cell instances, which have inferior drive capability.

Since the utilization of X1p25 and X1p5 drives did not increase, despite the fact that the “cost function” of the next level of drive strength (X2) was scaled to be more “expensive”, leads to the conclusion that there is little benefit in providing cells with too many drive instances.

Library	KCELL.P1	KCELL.P2	KCELL.P3	KCELL.P4
Total No. of Cells	7802	7610	8185	7665
Drive				
X0P5 [%]		43	51	45
X0P75 [%]		13	14	11
X1 [%]	70	12	11	21 (*9)
X1P25 [%]		6	6	6
X1P5 [%]		1	2	2 (*1)
X2 [%]	25 (*11)	22 (*11)	14 (*10)	13 (*1)
Others [%]	5	3	2	2

Note: The (* “number”) notation represents the percentage of sequential cells (of the total design)

Table 6.11: Comparison of Drive Utilization: The A34 Mapped to Different Target Libraries (Using the LARGE wire load model)

6.4.2 Mapping to “kcell.p4”

The “kcell.p4” library version is identical to the “kcell.p2” version, except for the two additional drive instances of the D-type flip-flop: X1 and X1p5 (Section 4.5.1). The three benchmark designs were mapped to the kcell.p4 version, and the drive utilization results of the A34 are summarized in Table 6.11. More than 80% (9/11) of the X2 drive sequential cells (column KCELL.P2) were replaced by the new X1 drive instances (col-

umn KCELL.P4). Similar results were obtained for the other benchmarks as well.

6.5 Summary

The results show consistent power saving when using the kcell.p2 library as compared to kcell.p1. For small blocks, modeled with the **SMALL** wire load, the total current was reduced by 20%-30%. For larger blocks, modeled with the **MEDIUM** wire load, the current was lowered by 5%-15% in most cases. Using the **LARGE** wire load model resulted in a total saving of 2%-15%.

The X0p5 and X0p75 instances were most often selected during technology mapping (kcell.p2), replacing the X1 instances (kcell.p1). The majority of the power reduction can be attributed to this phenomenon.

The synthesis results using the kcell.p4 library version, indicate that additional power saving is feasible when the sequential cells are offered with multiple drive strengths.

Chapter 7

Conclusions

7.1 Summary

This thesis has focused on developing a low power standard cell library, containing multiple drive instances for its cells. The library consists of 248 cells, and includes models for a large variety of CAD tools. A major effort of several months has been spent on setting up a complex experimental infrastructure which allows the simulation of power in large circuits. This allowed the comparison of several standard cell libraries, and the assessment of their performance in terms of power dissipation.

The simulation results show that providing standard cell libraries with multiple drive instances is extremely important for minimizing power in synthesized designs: The total simulated current in all designs, using three different wire load models, was consistently reduced when mapped to the multiple drive library. Although not quantified, further reduction is expected if the sequential cells are provided with multiple drives as well.

The results obtained for the LARGE wire load model indicate that less power saving can be expected by this method when using more advanced technologies (0.5 μm , 0.35 μm) yet, it should be noticed that the library developed for this thesis was compared to a library that already had several drive strength levels for each cell, thus the benefits are expected to be much larger in comparison to a library without various drive instances. Therefore, multiple drive libraries are still useful for deep sub-micron technologies.

Two different approaches for scaling the area attributes were investigated. For both approaches, the cells with the smallest area were most often selected during technology mapping. It was also found that specific drive instances are rarely selected, and there is little benefit in keeping them in the library.

7.2 Contributions

This thesis makes two important contributions which allow the minimization of power in synthesized designs:

1. A low power, standard cell library used for telecom applications has been further improved by providing its cells with additional drive instances. It was shown that the utilization of the modified library yields a reduction of 2%-15% (minimum) of the total power dissipation.
2. An experimental platform, including PowerMill and Vertue, has been put in place and integrated into one of NT's digital design flows for the 0.8 μm BATMOS process. This platform is useful not only for comparison between libraries, but also for simulating the power dissipation in large IC designs. Designers can use this platform to obtain power information for a specific module during regular logic simulation.

The same experimental platform can be used for future libraries developed for this process. With minor modifications, it could be used for other technologies as well.

7.3 Future Research

Since the experimental infrastructure can be easily modified to support new cells in the library, additional functions can be added to the multiple drive library, for example a variety of complex gates. Their relative contribution in reducing the power can be

estimated thereafter.

The utilization of the X1p25 and especially the X1p5 drives was very low, therefore it should be further researched whether having only one stage of drive strength between X1 and X2 would result in better utilization and improved performance (power).

Although the LARGE wire load model was used to assess the merits of multiple drive cells in deep sub-micron technologies, better results could be obtained if similar experiments are carried out using 0.5 μm or even 0.35 μm technologies.

The same experimental procedure can be followed once again, with one difference: Instead of using wire load models as optimization constraints for the synthesis tool, parasitic capacitances derived from the physical layout can be back annotated, and provided as environmental constraints for a second phase of synthesis. This would be the optimal way to obtain accurate synthesis results, since the physical characteristics of the design are then taken into consideration.

A similar platform can be used to evaluate the performance of future standard cell libraries, such as libraries using low threshold voltages or complex functions implemented with transmission gates (pass transistors).

Glossary of Terms

Analog Artist	A design framework CAD tool, used for IC design (from Cadence).
ACCELL	A BNR/NT proprietary software for cell characterization.
ASIC	Application Specific Integrated Circuit
BATMOS	A 0.8 micron IC technology process.
BNR	Bell Northern Research.
CAD	Computer Aided Design.
Characterization	Is the process in which the performance of the cells in a standard cell library is evaluated through simulations. Rise/Fall times, propagation delays, input cap. and drive capability are the common values which are derived for each cell in the library.
CMOS	Complementary Metal Oxide Semiconductor.
Design Analyzer	Is a GUI to the various Synopsys synthesis tools. Most of the synthesis capabilities are directly accessible from Design Analyzer menus.
Design Compiler	Is part of the Synopsys synthesis tool. Creates an optimized gate-level implementation of a given HDL design.
Gate Ensemble	A P&R tool (from Cadence).
gentech	A utility program within the PowerMill software package that

	creates the “Epic Technology File”.
HDL	Hardware Description Language.
HDL Compiler	Is part of the Synopsys synthesis tool. Reads a given HDL design, and compiles it to an internal data-base format.
HLS	High Level Synthesis.
HSPICE	A commonly used transistor level circuit simulator (from Meta Software).
IC	Integrated Circuit.
kcell	The name of Northern Telecom’s 0.8 μm standard cell library.
Logic Synthesis	The process of generating a gate level netlist based on the HDL design and a technology specific target library.
MOS	Metal Oxide Semiconductor.
Multiple Drive Cell	A cell in the library which has several instances (Drives). The instances differ in their speed and power.
NMOS (nmos)	N-type MOS transistor.
NT	Northern Telecom.
PMOS (pmos)	P-type MOS transistor.
PowerMill	An event driven, transistor level simulator (from EPIC Designs Inc.).
P & R	Place and Route.
spice2e	A utility program within the PowerMill software package that translates HSPICE netlists into Epic format.
Synopsys	Is a synthesis CAD tool (Synopsys Inc.). Can read HDL

designs written in Verilog or VHDL and creates an optimized gate level implementation (from Synopsys Inc.).

Test Bench

A file that contains simulation vectors.

Verilog

Is a hardware description language. There is also a logic simulator with this name (Both products are from Cadence).

Vertue

An interface software between Verilog and PowerMill.

VLSI

Very Large Scale Integrated circuit.

List of Symbols

C_{ox}	Thin oxide capacitance. The units are “capacitance per unit area”.
F_{ck}	The system’s clock frequency.
$I_{avV_{dd}}$	Total average current consumed from the Vdd rail.
L	The length of the transistor’s channel.
t_{PLH}	The “low” to “high” propagation delay.
t_{PHL}	The “high” to “low” propagation delay.
V_t	Threshold voltage.
V_{in}	Threshold voltage of n-type transistors.
V_{tp}	Threshold voltage of p-type transistors.
W	Transistor’s channel width.
W_n	n-type transistor’s channel width.
W_p	p-type transistor’s channel width.
wt	Watts.

Appendix A: Multiple Drive Library Listing

This documentation as well as additional data can be found at doe.carleton.ca under the following directories:

Root Directory:	~/tmp_mnt/home/
This document in “Frame-Maker” format:	Root/hronny/public/thesis/DOCS/
“Perl” executables:	Root/hronny/public/thesis/PERL/
Epic-BATMOS “Tech” file:	Root/hronny/public/thesis/EPIC/
Verilog files:	Root/hronny/public/thesis/VERILOG/

Information on the specific files is provided in “README” files in each one of the corresponding directories.

This appendix contains the listing of the drive instances that were created for the modified multiple drive library. Together with the reference library, kcell.p1, they formed the kcell.p2 library.

Cell Name	Functionality Description	Drive Instance	Max. Intrinsic Delay* [ns]	Maximum Output Drive** [pf]
kand2	2 Input AND	X0p5	0.933	0.0957
		X0p75	0.827	0.1555
		X1p25	0.745	0.2734
		X1p5	0.838	0.3303

Cell Name	Functionality Description	Drive Instance	Max. Intrinsic Delay* [ns]	Maximum Output Drive** [pf]
kand3	3 Input AND	X0p5	1.417	0.0906
		X0p75	1.184	0.1495
		X1p25	0.988	0.1607
		X1p5	1.311	0.3175
kaoi211	$F=(AB+C+D)'$	X0p75	1.285	0.0198
		X1p25	1.068	0.0489
		X1p5	1.041	0.0611
kaoi21	$F=(AB+C)'$	X0p5	0.773	0.0316
		X0p75	0.688	0.0538
		X1p25	0.618	0.1004
		X1p5	0.616	0.1208
kaoi22	$F=(AB+CD)'$	X0p5	1.174	0.0208
		X0p75	1.032	0.0389
		X1p25	0.941	0.0733
		X1p5	0.939	0.0886
kaoi31	$F=(ABC+D)'$	X0p5	0.948	0.0267
		X0p75	0.842	0.0472
		X1p25	0.779	0.0842
		X1p5	0.767	0.1025
kbfb	Buffer	X0p5	0.727	0.0983
		X0p75	0.627	0.1582
		X1p25	0.568	0.2792
		X1p5	0.577	0.3358
kiv	Inverter	X0p5	0.212	0.0928
		X0p75	0.192	0.1496
		X1p25	0.183	0.2628
		X1p5	0.177	0.3186
kmux2	2 Input 1 Select MUX	X0p5	1.288	0.0948
		X0p75	1.288	0.1407
		X1p25	1.134	0.2332
		X1p5	1.122	0.2685

Cell Name	Functionality Description	Drive Instance	Max. Intrinsic Delay* [ns]	Maximum Output Drive** [pf]
kmuxi2	2 Input 1 Select Inverting MUX	X0p5	1.064	0.0319
		X0p75	1.118	0.0425
		X1p25	1.171	0.0897
		X1p5	1.101	0.2776
knd2	2 Input NAND	X0p5	0.362	0.0701
		X0p75	0.322	0.1169
		X1p25	0.292	0.2120
		X1p5	0.287	0.2573
knd3	3 Input NAND	X0p5	0.512	0.0478
		X0p75	0.447	0.0789
		X1p25	0.412	0.1469
		X1p5	0.413	0.1818
kndi2	$F=[(A')B]'$	X0p5	0.877	0.0698
		X0p75	0.729	0.1166
		X1p25	0.638	0.2107
		X1p5	0.624	0.2626
knr2	2 Input NOR	X0p5	0.523	0.0371
		X0p75	0.458	0.0628
		X1p25	0.415	0.1136
		X1p5	0.410	0.1355
knr3	3 Input NOR	X0p75	0.839	0.0342
		X1p25	0.754	0.0651
		X1p5	0.739	0.0797
knri2	$F=(A'+B)'$	X0p5	0.918	0.0387
		X0p75	0.783	0.0642
		X1p25	0.703	0.1148
		X1p5	0.691	0.1392
koai211	$F=[(A+B)CD]'$	X0p5	1.087	0.0187
		X0p75	0.937	0.0366
		X1p25	0.865	0.0693
		X1p5	0.851	0.0848

Cell Name	Functionality Description	Drive Instance	Max. Intrinsic Delay* [ns]	Maximum Output Drive** [pf]
koai21	$F=[(A+B)C]'$	X0p5	0.816	0.0292
		X0p75	0.722	0.0508
		X1p25	0.647	0.0952
		X1p5	0.644	0.1144
koai22	$F=[(A+B)(C+D)]'$	X0p5	0.838	0.0240
		X0p75	0.752	0.0437
		X1p25	0.684	0.0820
		X1p5	0.668	0.0991
koai31	$F=[(A+B+C)D]'$	X0p5	1.253	0.0136
		X0p75	1.129	0.0254
		X1p25	1.024	0.0504
		X1p5	0.994	0.0633
kor2	2 Input OR	X0p5	0.901	0.0976
		X0p75	0.728	0.1566
		X1p25	0.664	0.275
		X1p5	0.667	0.3368
kor3	3 Input OR	X0p5	1.07	0.0906
		X0p75	1.01	0.1561
		X1p25	0.902	0.2715
		X1p5	0.757	0.3333
kxnr2	2 Input Inverting XOR	X0p5	2.321	0.0851
		X0p75	2.363	0.1357
		X1p25	2.018	0.2491
		X1p5	1.99	0.304
kxor2	2 Input XOR	X0p5	2.318	0.0863
		X0p75	2.356	0.1373
		X1p25	2.016	0.2489
		X1p5	1.943	0.3039

* Specifies the worst case intrinsic delay (rise or fall) from an input pin to the output.

** Specifies the maximum capacitive load that can be driven at the output, such that the rise/fall time is less than 3ns.

Appendix B: Synopsys Library Models

This appendix presents the Synopsys “Technology Library” format, including partial information on the “MEDIUM” wire load model. Several cells from the multiple drive library (kcell.p2) are shown as well. Additional data regarding the Synopsys library formats can be found in [30].

```
library(kcell) {
```

The following are the general “library attributes”, including specifications of the default values.

```
date      : "Mon Feb 27 16:51:27 1995" ;
time_unit  : "1ns" ;
voltage_unit : "1V" ;
current_unit : "1mA" ;
pulling_resistance_unit : "1kohm" ;
capacitive_load_unit(1,pf);

default_output_pin_fall_res : 0.0
default_slope_rise          : 0.0
default_fanout_load         : 1.0
default_inout_pin_fall_res  : 0.0
default_intrinsic_fall     : 1.0
default_intrinsic_rise      : 1.0
default_output_pin_rise_res : 0.0
default_output_pin_cap      : 0.0
default_input_pin_cap       : 1.0
default_inout_pin_rise_res  : 0.0
default_slope_fall          : 0.0
default_inout_pin_cap       : 1.0

/* wireload models - the units for length is microns*/
/* second-level metal parameters are used */
/*****/
/* Wireload file for synopsys. */
/* Parameters used: */
/* Log-Linear Regression */
/* point estimator: mean + 0.00 std-deviations */
/* Generation time: Tue Feb 21 14:29:07 EST 1995 */
/* 50 fanouts are defined below. */
/* This wireload file is intended for use for */
/* "medium" size blocks. */
/*****/

wire_load("wireload.MEDIUM") {
    resistance = 0 ;
```

```

    capacitance = 1 ;
    area = 0 ;
    slope : 1.000000;
    fanout_length( 1, 0.008624);
    fanout_length( 2, 0.022442);
    fanout_length( 3, 0.057509);
    fanout_length( 4, 0.069250);
    fanout_length( 5, 0.080854);
    fanout_length( 6, 0.092325);
    fanout_length( 7, 0.103666);
    fanout_length( 8, 0.114880);
    fanout_length( 9, 0.125970);
    fanout_length(10, 0.136938);

    fanout_length(49, 0.491398);
    fanout_length(50, 0.499021);

}

```

The following is an example of the “cell description” format, and the related attributes that are taken from the characterization data:

```

cell( kiv_0p5 ) {
    area : 217 ;

    pin( a ) {
        direction : input ;
        capacitance : 0.0111 ;
    }

    pin( opb ) {
        direction : output ;
        max_transition : 1.1368 ;
        function : “!a” ;
        timing() {
            intrinsic_rise : 0.2124 ;
            slope_rise : 0.255278 ;
            rise_resistance : 12.2 ;
            intrinsic_fall : 0.2767 ;
            slope_fall : 0.321846 ;
            fall_resistance : 12.25 ;
            related_pin : “a” ;
        }
    }
}

cell( knd2_0p5 ) {
    area : 325 ;

    pin( a1 ) {
        direction : input ;
        capacitance : 0.0124 ;
    }

    pin( a2 ) {

```



```

    direction : input ;
    capacitance : 0.0113 ;
}

pin( opb ) {
    direction : output ;
    max_transition : 1.14053 ;
    function : "!( a1 & a2 )";
    timing() {
        intrinsic_rise : 0.2969 ;
        slope_rise : 0.258 ;
        rise_resistance : 14.88 ;
        intrinsic_fall : 0.3628 ;
        slope_fall : 0.275692 ;
        fall_resistance : 16.27 ;
        related_pin : "a1" ;
    }
    timing() {
        intrinsic_rise : 0.3628 ;
        slope_rise : 0.275944 ;
        rise_resistance : 14.73 ;
        intrinsic_fall : 0.3793 ;
        slope_fall : 0.192667 ;
        fall_resistance : 15.94 ;
        related_pin : "a2" ;
    }
}
}

```

Note: The following are the minimum requirements to form a complete "Technology Library":

- 1) At least one inverter cell.
- 2) Either a two input NOR gate, or both a two input AND gate and a two input OR gate.

During technology mapping, any boolean function can be realized by a combination of these cells.

Appendix C: The Verilog Description of the Data-Path

```
module datapath ( out2, A, B, C, cin, clk);
input [3:0] A, B, C;
input cin;    // This is the carry_in of the fulladder.
input clk;
output [1:0] out2;

wire [3:0] A, B, C;
wire cin, clk;
wire [1:0] out2;

wire [3:0] qa;
wire [3:0] qb;
wire [3:0] qc;
wire [3:0] sumfa;

latch a (qa, clk, A),
      b (qb, clk, B),
      c (qc, clk, C);

fulladder f1 (coutfa, sumfa, qa, qb, cin); // qa qb instead of A, B.

comparator c1 ( out2, coutfa, sumfa, qc); //qc instead of C
endmodule
```

```
module latch (q, clock1, d); // 4 bit data latch
```

```
input clock1;
input [3:0] d ;
output [3:0] q ;
reg [3:0] q;
```

```
wire [3:0] d;
wire clock1;
```

```
always @ (clock1 or d)
begin
```

```
if ( clock1 )
begin
```

```
    q = d;
end
end
endmodule
```

```
module fulladder (carry_out, sum, x, y, carry_in);
```

```
output carry_out;
output [3:0] sum;
input [3:0] x;
input [3:0] y;
input carry_in;

wire [3:0] sum;
wire carry_out;
wire [3:0] x, y;
wire carry_in;

assign {carry_out, sum} = x + y + carry_in;

endmodule

// This is a modified version of the comparator to allow handling
// of the "carry out" from the fulladder.

module comparator (out3, cina, a, b);

output [1:0] out3;
input [3:0] a, b;
input cina; // This is the carry_out from the fulladder.

reg [1:0] out3;

always @ (a or b or cina)
begin
    if (cina == 0) // Carry_out of the fulladder is zero.
    begin
        if (a > b)
            out3 = 2'b01;
        else if (a == b)
            out3 = 2'b00;
        else if (a < b)
            out3 = 2'b11;
        end
    else if (cina == 1)
        out3 = 2'b01;
    end
endmodule
```

References

- [1] Anantha P. Chandrakasan, Samuel Sheng and Robert W. Broderson, "Low Power CMOS Digital Design", *IEEE Journal of Solid State Circuits*, Vol. 27, No. 4, pp. 473-484, April 1992.
- [2] Dake Liu and Christer Svensson, "Trading Speed for Low Power by Choice of Supply and Threshold Voltages", *IEEE Journal of Solid State Circuits*, Vol. 28, No. 1, pp. 10-17, Jan. 1993.
- [3] Mark Horowitz, Thomas Indermaur, and Ricardo Gonzalez, "Low Power Digital Design", *Proc. IEEE Symposium on Low Power Electronics*, pp. 8-11, 1994.
- [4] F. Dresig, Ph. Lanches, O. Rettig, and U. G. Baitinger, "Simulation and Reduction of CMOS Power Dissipation at Logic Level", *Proc. IEEE DAC*, pp. 341-346, 1993.
- [5] Kurt Keutzer and Peter Vanbekbergen, "The Impact of CAD on the Design of Low Power Digital Circuits", *Proc. IEEE Symposium on Low Power Electronics*, pp. 42-45, 1994.
- [6] Kurt Keutzer and Ken Scott, "Improving Cell Libraries for Synthesis", *Custom Integrated Circuits Conference*, pp. 1-11, 1994.
- [7] Vivek Tiwari, Pranav Ashar, and Sharad Malik, "Technology Mapping for Low Power", *Proc. IEEE DAC*, pp. 74-79, 1993,.
- [8] Chi-Ying Tsui, Massoud Pedram, Alvin M. Despain, "Technology Decomposition and Mapping Targeting Low Power", *Proc. IEEE DAC*, pp. 68-73, 1993.

- [9] Bernhard Hoppe, Gerd Neuendorf, Doris Schmitt, "Optimization of High-Speed Logic Circuits with Analytical Models for Signal Delay, Chip Area, and Dynamic Power Dissipation", *IEEE Transaction on Computer Aided Design*, Vol. 9, No. 3, pp. 236-247, March 1990.
- [10] Kerry S. Lowe and P. Glenn Gulak, "Gate Sizing and Buffer Insertion for Optimizing Performance in Power Constrained BiCMOS Circuits", *IEEE Journal of Solid State Circuits*, Vol. 28, No. 1, pp. 216-219, Jan. 1993.
- [11] M. Tachibana, S. Kurosawa, R. Nojima, "Power and Area Optimization by Reorganizing CMOS complex Gate Circuits" *Proc. ISLPD Symposium*, pp. 155-160, 1995.
- [12] C. Piguet, J-M. Masgonty, V. von Kaenel, "Logic Design for Low-Voltage/Low Power CMOS Circuits", *Proc. ISLPD Symposium*, pp. 117-12, 1995.
- [13] Takeshi Tokuda, Tohru Kengaku, Eiichi Teraoka, "A Mixed Signal DSP for Single-Chip Speech Codec", *IEICE Transaction on Electronics*, Vol. E75-C, No. 10, pp. 1241-1247, October 1992.
- [14] Anthony Correale, Jr. "Overview of the Power Minimization Techniques Employed in the IBM PowerPC 4xx Embedded Controllers", *Proc. ISLPD Symposium*, pp. 75-80, 1995.
- [15] Hiroaki Kaneko, Takashi Miyazaki, Hideki Sugimoto, "A Design of Static Operatable Low-Power 16-bit Microprocessor", *IEICE Transaction on Electronics*, Vol. E75-C, No. 10, pp. 1188-1195, October 1992.
- [16] Kenneth J. Schultz, Robert G. Gibbins, James S. Fujimoto, "Low-Supply-Noise Low-Power Embedded Modular SRAM for Mixed Analog-Digital ICs", *IEEE Proc. Custom*

Integrated Circuits Conference, pp. 7.1.1-7.1.4, 1992.

[17] Katsuhiro Shimohigashi and Koichi Seki, "Low Voltage ULSI Design", *IEEE Journal of Solid State Circuits*, Vol. 28, No. 4, pp. 408-413, April 1993.

[18] Zongjian Cchen, John Shott, James Burr, "CMOS Technology Scaling for Low Voltage Low Power Applications", *Proc. IEEE Symposium on Low Power Electronics*, pp. 56-57, 1994.

[19] P. H. Woerlee, C. A. H. Juffermans, H. Lifka, "A Low Power 0.25 um CMOS Technology", *Proc. IEEE IEDM*, pp. 2.4.1-2.4.4, 1992.

[20] Kimiyoshi Usami and Mark Horowitz, "Clustered Voltage Scaling Technique for Low-Power Design", *Proc. ISLPD Symposium*, pp. 3-8, 1995.

[21] Salil Raje and Majid Sarrafzadeh, "Variable Voltage Scheduling", *Proc. ISLPD Symposium*, pp. 9-14, 1995.

[22] Laurence Goodby, Alex Orailoglu, Paul M. Chau, "A High Level Synthesis Methodology for Low-Power VLSI Design", *Proc. IEEE, Symposium on Low Power Electronics*, pp. 48-49, 1994.

[23] E. Musoll and J. Cortadella, "High Level Synthesis Techniques for Reducing the Activity of Functional Units", *Proc. ISLPD Symposium*, pp. 99-104, 1995.

[24] Luca Benini and Giovanni De Micheli, "Transformation and Synthesis of FSM for Low-Power Gated-Clock Implementation", *Proc. ISLPD Symposium*, pp. 21-26, 1995.

[25] Christopher K. Lennard and A. Richard Newton, "An Estimation to Guide Low

Power Resynthesis”, *International Symposium on Low Power Design*, pp. 227-232, 1995.

[26] Christos Papachristou, Mark Spining, Mehrdad Nourani, “A Multiple Clocking Scheme for Low-Power RTL Design”, *Proc. ISLPD Symposium*, pp. 27-32, 1995.

[27] Vivek Tiwari, Sharad Malik, Pranav Ashar, “Guarded Evaluation: Pushing Power Management to Logic Synthesis/Design”, *Proc. ISLPD Symposium*, pp. 70-75, 1995.

[28] Charlie X. Huang, Bill Zhang, An-Chang Deng, “The Design and Implementation of PowerMill”, *Proc. ISLPD Symposium*, pp. 105-109, 1995.

[29] Neil Weste, Kamran Eshraghian, “Principles of CMOS VLSI Design, A Systems Perspective”, *Addison Wesley*, 1988.

[30] Tom Burd, “Low Power CMOS Library Design Methodology”, *Master Thesis, University of California, Berkley*, 1995.

[31] Synopsys “Library Compiler”, *Reference Manual*, October 1992.

[32] EPIC “PowerMill”, *Reference Manual*, 1994.

IMAGE EVALUATION TEST TARGET (QA-3)

APPLIED IMAGE, Inc.
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989

© 1993, Applied Image, Inc., All Rights Reserved