

HIERARCHICAL POWER OPTIMIZATION FOR ULTRA LOW-POWER DIGITAL SYSTEMS

A Thesis
Presented to
The Academic Faculty

By

Kyu-Won Choi

In Partial Fulfillment of the Requirements for
the Degree Doctor of Philosophy
in Electrical and Computer Engineering




Georgia Institute of Technology

October 16, 2003

Copyright © 2003 by Kyu-Won Choi

HIERARCHICAL POWER OPTIMIZATION FOR ULTRA LOW-POWER DIGITAL SYSTEMS

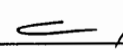
Approved:



Abhijit Chatterjee, Chairman

Madhaven Swaminathan

SungKyu Lim



Gabriel Rincón-Mora

YingJie Liu

Date approved by chairman: 10-15-03

DEDICATION

To my family:

My wife, JooHee Kim

and

My son, WonJong (Alex) Choi

For their love, encouragement, understanding and believing.

ACKNOWLEDGMENT

First and foremost, I would like to express my deep appreciation to my academic advisor, Dr. Abhijit Chatterjee, for his valuable guidance, support and encouragement during my graduate study at Georgia Tech. I am grateful to him for giving me an opportunity to study in low-power design techniques for future generation systems and for teaching me how to pursue aggressive research in this field. His endless and invaluable suggestions are a great aid to this research work.

I would like to thank my reading committee members, Dr. Gabriel Rincón-Mora, Dr. Sung-Kyu Lim, for their efforts in reviewing this PhD. Dissertation and providing constructive comments. I would also like to express my thanks to Dr. Madhavan Swaminathan and Dr. YingJie Liu for their constructive comments and supports as my Ph.D. committee members. I am thankful to my colleagues Dong, Andy, Yuvraj, Utku, Achintya, Soumendu, Sermet, Donghoon and Ganesh for the company and valuable discussions.

Most importantly, I would like to express my gratitude to my wife, JooHee Kim, for her endless love, support and endurance, and to my family; my grand father, YangSoon Choi, my parents, SungHa Choi and SoonOhk Lee, and my sisters, KwangHee Choi and YounHee Choi, for endless love, support, and encouragement. I would like to thank to my elder sister's husband, Dr. ChongDu Cho, for his valuable advice and

encouragement. Last, but not least, I would like to express my gratitude to my son, WonJong Choi, for encouraging me with his invaluable smile during the PhD. work.

TABLE OF CONTENTS

Dedication	iii
Acknowledgment	iv
Table of Contents	vi
List of Figures	ix
List of Tables	xiii
Summary	xiv
CHAPTER 1	1
Introduction	1
1.1 Background	3
1.2 Key Contributions and Research Scope	9
1.3 Overview of the dissertation	10
CHAPTER 2	14
Prior Work and Proposed New Design Flow	14
2.1 Sources of Power Dissipation	16
2.2 Summary of Low-Power Techniques	18
2.3 Energy and Delay Tradeoffs	22
2.4 Previous Work	23
2.4.1 Instruction Scheduling	23
2.4.2 Timing Analysis	25
2.4.3 Supply Voltage Scaling	26
2.4.4 Threshold Voltage Scaling	27
2.4.5 Gate Sizing	29
2.4.6 Interconnect Scaling	30
2.5 The New Design Flow	31
CHAPTER 3	33
Low-Power Instruction Scheduling	33
3.1 Introduction	33
3.2 Problem Description	33
3.3 Instruction-Level Power Optimization Algorithm	36
3.3.1 CDG Construction	38
3.3.2 PDT Generation	39
3.3.3 SCG Generation	40
3.3.4 TSP Implementation	41
3.4 Validation of the Optimization	44
3.5 Experimental Results	47

3.6 Summary.....	50
CHAPTER 4.....	51
Graph-Based Timing Analysis for Minimum Power.....	51
4.1 Introduction	51
4.2 Graph-Based Timing Analysis	53
4.3 Energy and Delay Model.....	56
4.4 Optimal Slack Distribution for Minimum Power	58
4.4.1 Previous Approaches	58
4.5 Power Aware Zero Slack Algorithm (PA-ZSA).....	61
4.5.1 Algorithm Description.....	61
4.5.2 Comparison with ZSA and MISA	70
4.5.3 Time Complexity Comparison	78
4.6 Gate-Level Power Optimization.....	78
4.7 Experimental Results.....	80
4.8 Summary.....	83
CHAPTER 5.....	84
Hierarchical Activity-Aware Delay Assignment for Low Power	84
5.1 Introduction	84
5.2 Hierarchical Activity-Aware Delay Assignment.....	85
5.2.1 Topological Depth-Based Partitioning	87
5.2.2 Activity-Aware Delay Assignment	88
5.3 Experimental Results.....	93
5.4 Summary.....	94
CHAPTER 6.....	96
Ultra Deep-Sub-Micron (UDSM)-Aware Gate-Level Power Optimization for Low-Power CMOS VLSI.....	96
6.1 Introduction	96
6.2 UDSM-Aware Design	98
6.3 Delay and energy model.....	101
6.3.1 MOSFETs in UDSM	101
6.3.2 Interconnects in UDSM.....	102
6.3.3 Component Delay (Gate+Interconnect) Model	102
6.3.4 Component Energy (Gate+Interconnect) Model	103
6.4 UDSM-Aware Post-Layout Power Optimization.....	103
6.4.1 Pre-procedure.....	106
6.4.2 Proposed Heuristic.....	108
6.5 Experimental results	112
6.6 Summary.....	119
CHAPTER 7.....	120
Conclusion.....	120
7.1 Research Findings.....	120
7.2 Future Directions	126

Bibliography.....	127
Vita.....	146

LIST OF FIGURES

Figure 1. Slack map for benchmark circuits.....	4
Figure 2. Slack distribution for 12 benchmark circuits.	5
Figure 3. Activity profiling.....	6
Figure 4. Power optimization rationale.	8
Figure 5. Overview of the proposed approaches.	12
Figure 6. Improvement in technology [2-8].	15
Figure 7. Sources of power dissipation.	16
Figure 8. Low-power techniques in methodology view.....	21
Figure 9. Energy and delay tradeoffs for power optimization.....	23
Figure 10. Power-performance tradeoffs for supply voltage scaling [2-26].	27
Figure 11. Proposed design flow.	32
Figure 12. Pre-procedure for instruction scheduling.....	34
Figure 13. CDG and search space example.....	35
Figure 14. Proposed instruction-level optimization approach.....	37
Figure 15. CDG construction algorithm.....	39
Figure 16 PDT generation example.....	40
Figure 17. SCG generation algorithm.....	41
Figure 18 Prim's algorithm example for MST.	42
Figure 19 2-interchange local Search heuristics.....	43

Figure 20 3-interchange local search heuristics.	43
Figure 21 The rank distribution for a sample of size 10.....	46
Figure 22. Scheduling example for one basic block of FIR source.....	48
Figure 23. Power savings for each component	49
Figure 24. Arrival time.	53
Figure 25. Required time.	54
Figure 26. Graph-based timing symbol.	55
Figure 27. Procedure of ZSA.....	59
Figure 28. Procedure of MISA.	60
Figure 29. Proposed PA-ZSA algorithm.	63
Figure 30. Aggishn_EDR algorithm.....	64
Figure 31. Two nodes example.....	66
Figure 32. Two modules example.	69
Figure 33. EDR validation.....	69
Figure 34. An example circuit (c17).....	70
Figure 35. DAG mapping.	71
Figure 36. Timing and slack analysis.	71
Figure 37. Slack-sensitive paths.	72
Figure 38. Slack distribution of ZSA.....	74
Figure 39. Slack distribution of MISA.	76
Figure 40. Slack distribution of PA-ZSA.	77
Figure 41. Gate-level power optimizer.....	80

Figure 42. Optimization procedure.....	81
Figure 43. Hierarchical delay assignment and power optimization.	85
Figure 44. Power optimization overview.	86
Figure 45. Partitioning overview.	87
Figure 46. Partitioning algorithm.	88
Figure 47. An example module-level circuit.	89
Figure 48. Module-level delay assignment (step 1).	90
Figure 49. Module-level delay assignment (step 2).	90
Figure 50. Module-level delay assignment (step 3).	91
Figure 51. Module-level delay assignment (step 4).	91
Figure 52. Module-level delay assignment (step 5).	92
Figure 53. Hierarchical delay assignment algorithm.....	93
Figure 54. UDSM Effects for 12-Banchmark Circuits.....	99
Figure 55. Proposed power optimization design flow.....	104
Figure 56. Hierarchical delay decomposition.....	105
Figure 57. Pre-procedure for c499.....	108
Figure 58. Power and delay basis for an example circuit in Figure 57 (b) without optimization.....	109
Figure 59. Technology mapping for minimum power.	110
Figure 60. Violation paths.	110
Figure 61. After optimization.	112
Figure 62. Average power savings in different partitioning.....	116

Figure 63. Average time savings in different partitioning..... 117

Figure 64. Average power savings in different hierarchies..... 117

Figure 65. Average time savings in different hierarchies..... 118

Figure 66. Relative performance. 118

Figure 67. New Design Flow..... 124

LIST OF TABLES

Table 1. Research scope.	9
Table 2. Progress of technology and change of applications [2-1].	15
Table 3. Low-power techniques in design hierarchical view.	19
Table 4. Previous results for instruction-level power optimization.....	25
Table 5. Previous results for supply voltage scaling.	27
Table 6. Previous results for threshold voltage scaling.	29
Table 7. Previous results for gate sizing.....	30
Table 8. Total power saving results for some applications.	49
Table 9. Power budget comparison.	78
Table 10. Complexity comparison.....	78
Table 11. Results of PA-ZSA based optimization.....	82
Table 12. Comparison with other ZSAs.	83
Table 13. Summary of Technology Parameters [6-1]	100
Table 14. Before Optimization	114
Table 15. After Optimization.....	114
Table 16. After Optimization (cont'd).....	115
Table 17. After Optimization (cont'd).....	115
Table 18. After Optimization (cont'd).....	116

SUMMARY

The objective of the dissertation is to develop a vertically integrated framework for software-hardware-technology co-optimization for low-power digital systems. Such co-optimization has been difficult due to the complexity of the optimization problem and the fact that the optimization parameters stretch across software, hardware and technology parameters. The proposed framework for power optimization in the dissertation shows that while energy saving is possible using software, hardware and technology optimization individually, much larger savings are possible if a vertically integrated power optimization approach across software, hardware and technology boundaries is undertaken. The key contribution of this thesis is the formulation of a hierarchical delay-driven power-optimization approach that can be applied to architectural modules as well as gate-level design. Using architectural-level studies of module usage and activity, it is shown how low-level physical design and technology parameters can be selected to minimize power consumption. The proposed work in this thesis will impact cross-disciplinary research in the areas of solid-state microelectronics, computer architecture and computer science.

CHAPTER 1

INTRODUCTION

Recent advances in wireless networking technology and the exponential development of semiconductor technology have introduced new challenges in the design of portable devices such as personal digital assistants (PDAs). Power optimization for these embedded systems and power constrained mobile computing devices is an active area of research that has received considerable attention in recent years. As the scale of integration expands, more transistors, faster and smaller than their predecessors, are being packed into smaller chip. The steady growth in clock frequency and processing capacity per chip has increased power dissipation dramatically. Overall technology characteristics of the 2001 Edition of the *International Technology Roadmap for Semiconductors (ITRS)* show that the total number of transistors integrated on a chip will exceed the human population by 2013 and will become larger than the total number of neurons in a human brain by 2015. This reflects the rapid progress of technology and the need for new design paradigms [1-1].

Historically, design priority in the semiconductor industry has been driven by the competitive market forces of simultaneously increasing integration and speed, and reducing chip area and cost. Higher integration and speeds have resulted in increased

power consumption and heat dissipation. The problem is that in the case of battery technology, there is no equivalent of Moore's Law that forecasts a doubling of the complexity of microelectronic chips every 18 months, and Gilder's Law, which theorizes a similar exponential growth in communication bandwidth. In contrast, battery technology has improved very slowly, and only a 20% improvement in capacity is expected over the next 10 years. Therefore, it is clear that power optimization is a major concern for designing future generation digital systems [1-2].

In this thesis, we propose an efficient approach to minimize total power (switching, short-circuit, and leakage power) without performance loss for ultra-low power CMOS circuits. We present a framework for a hierarchical delay-driven power-optimization approach. At software level, we minimize total activity of target system by using an efficient instruction scheduling mechanism. At module level, we hierarchically assign each module's delay for minimum power. Given minimized activity and optimal delay, at gate level, we combine supply/threshold voltage scaling, gate sizing, and interconnect scaling techniques for power optimization in the presence of back end (post-layout-based) ultra-deep-sub-micron (UDSM) technology effects. We have tested the proposed algorithms on a set of benchmark circuits and several building blocks of a synthesizable ARM core. The experimental results show that our polynomial-time solvable strategy achieves over an order of magnitude savings in total power without compromising performance.

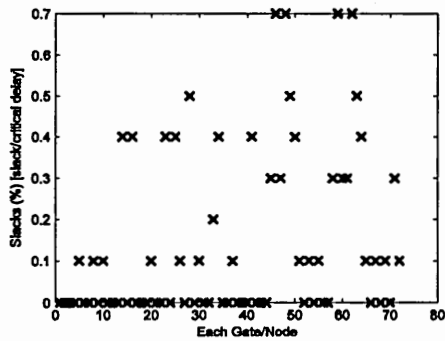
1.1 Background

The capability to fully exploit the signal processing power of the transistor was provided by the invention of the integrated circuit in 1958 [1-3]. Recently, there have been considerable interests in the design of power/energy-aware electronics seeking to reduce the power consumption of military and consumer applications by over two orders of magnitude [1-4~1-7]. This interest is driven by the power demands of today's high-speed, high-density, (portable) integrated electronics. Unless new power-efficient design methods are developed, there will be stringent limitations on the levels of electronics integration possible in the future as well as limitations on the speeds at which hardware can operate. Hardware portability will also be affected by battery size and weight. Simple calculations reveal that future microprocessors will dissipate several hundred watts of power unless drastic power reduction features are implemented. High power consumption and the associated high temperatures can also lead to reliability problems leading to early failure. If current trends in reduction of feature sizes and the increase in the levels of integration in silicon are to continue well into the future, the power consumption problem must be handled effectively and solved urgently.

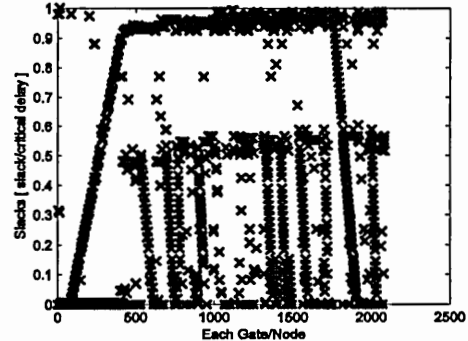
In general, low-power optimizations that do not compromise system performance are dependent on time slack calculation and the surplus delay (slack budget) distribution among the circuit modules. Time slack is measured as the difference between the signal required time and the signal arrival time at the primary output of each module. The first observation of this research is that the logic modules on non-critical paths in digital circuits typically have high timing slack which may allow the increase of their delay

without violating the timing constraints. Our experience shows that, in most circuits, there are few logic modules on critical paths of the total number of modules.

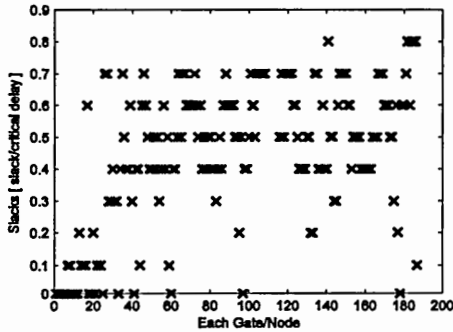
Figure 1 shows the slack map for several benchmark circuits. In these figures, the slack values (y-axis) have been normalized to the critical path delay. The y-axis shows slack ratio (slack / critical delay) in circuit and the x-axis represents each gate/module.



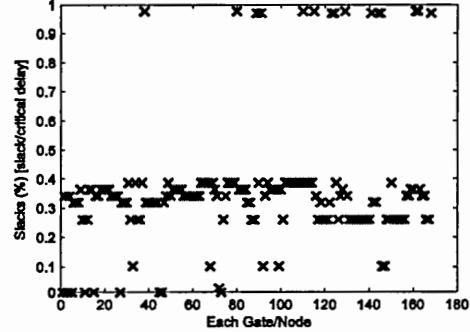
(a) 4-full adder



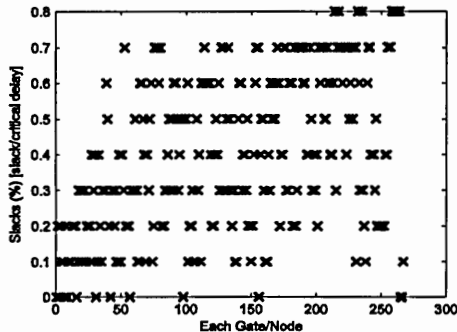
(b) 64-alu



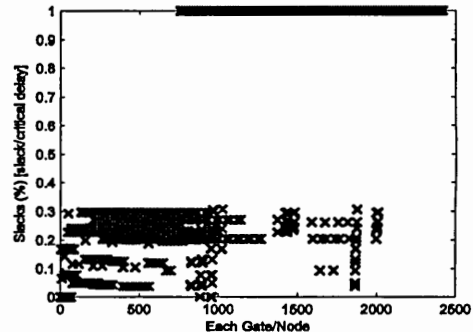
(c) s298



(d) s344



(e) s526



(f) c6288

Figure 1. Slack map for benchmark circuits.

As shown in Figure 2, after investigating 12-benchmarks after technology mapping, we found that the number of gates on critical paths accounts for only about 12 % of total gates, while more than 60 % of gates have their slack larger than 25% of the critical path delay. This potentially provides much room for power reduction through scaling of supply voltage, threshold voltage, and device/interconnects width.

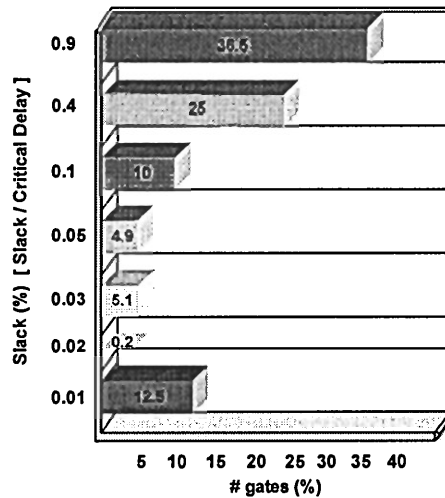


Figure 2. Slack distribution for 12 benchmark circuits.

The activity variation of each module/gate in digital circuits is shown in Figure 3. The activity is dependent on the topology of the circuit, the technology-mapping techniques, and the input vector.

Figure 3 shows activity profiles some benchmark circuits. *Monte Carlo simulation* is used to generate the activity profiles module/sub-module. This approach consists of applying randomly generated input patterns at the primary inputs of the circuit and monitoring the switching activity per time interval T using a simulator. Under the assumption that the switching activity of a circuit module over any period T has a normal

distribution, and for a desired percentage error in the activity estimate and a given confidence level, the number of required simulation vectors is estimated. The simulation based approach is accurate and capable of handling various device models, different circuit design styles, single and multi-phase clocking methodologies, tristate drives, etc.

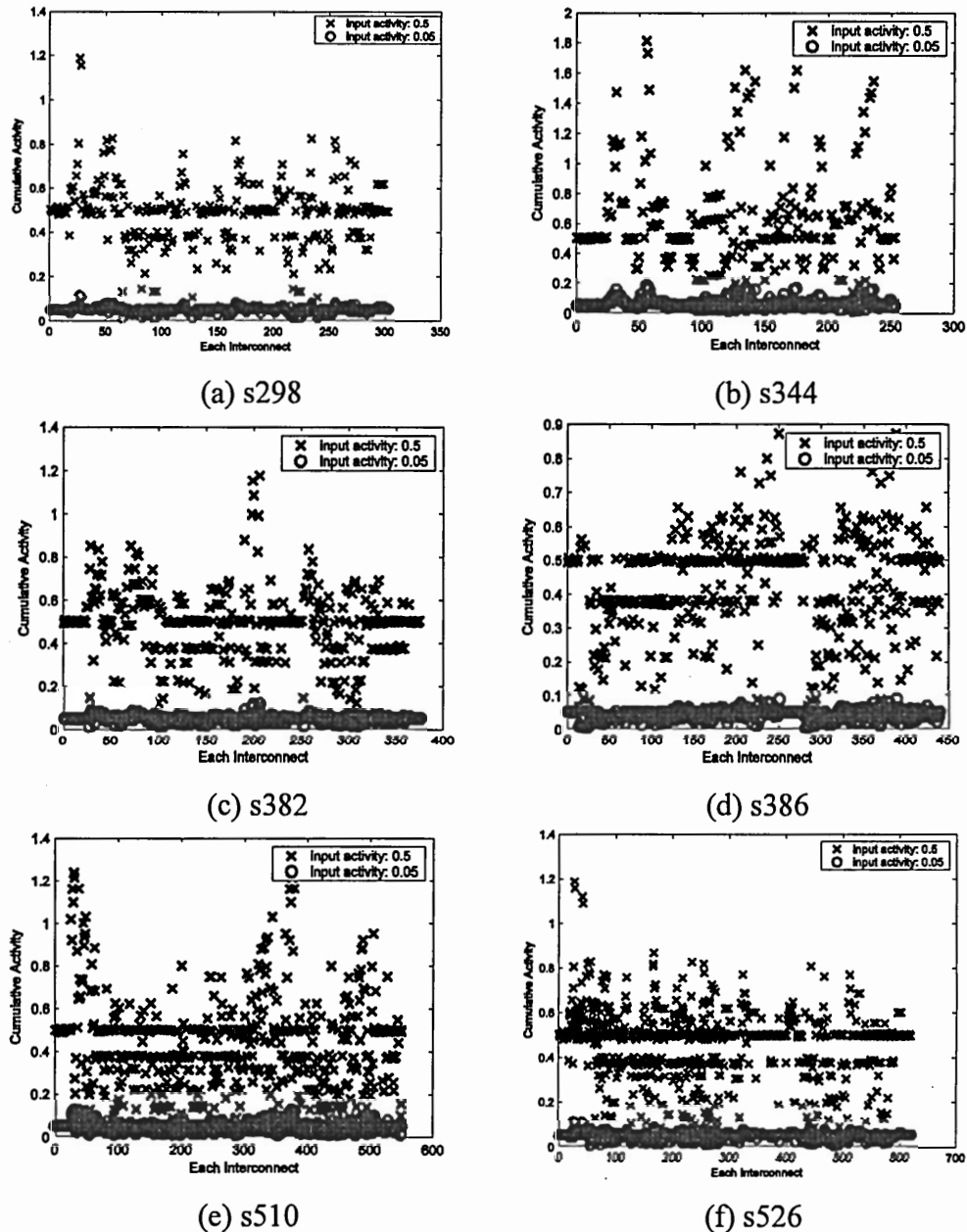
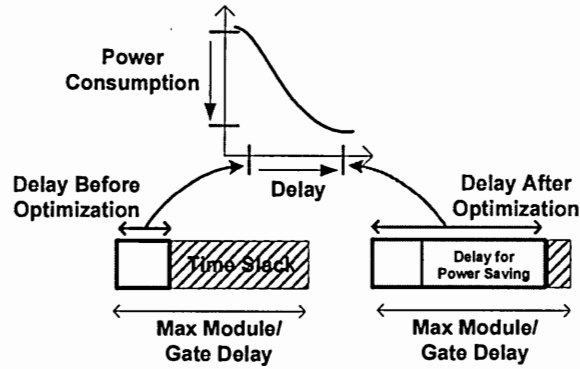


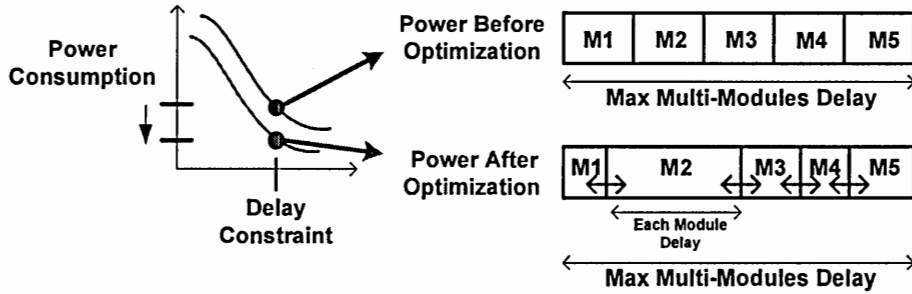
Figure 3. Activity profiling.

Figure 4 shows the rationale for module/gate level power optimization. The more the slack assigned to a module (a more power hungry module), the higher is the potential to save power by slowing down the module. This “slowing down” is achieved by optimally adjusting the modules’ supply voltage, threshold voltage and by resizing the gates and interconnections in the module to minimize total (dynamic and static) power.

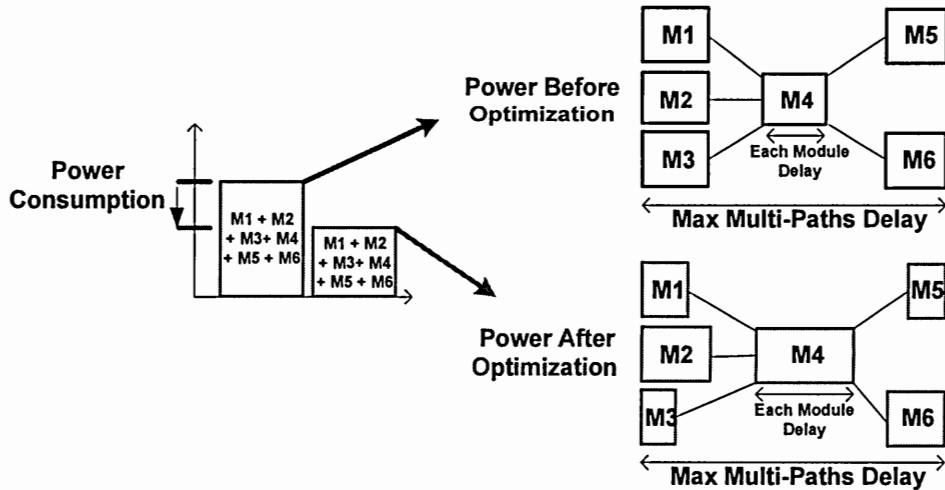
After critical path and slack analysis of the target circuit, for modules on non-critical path(s), power-optimal delay is assigned according to the rationale shown in Figure 4(a). Figure 4(b) shows that the delays on critical path (no slack on the path) can be readjusted for minimizing power. The Figure 4(c) shows the power optimization rationale for multiple modules on multiple paths. The problems of the delay assignment algorithms are the complexity due to the exponentially increased the number of paths and the non-linearity of the trade-offs among the multiple technology parameters. To handle the complexity problem, we propose a simple optimization metric. By using such a metric, the problem of technology optimization (V_{dd} , V_{th} , W_{gate} , and W_{int} scaling) can be decoupled from that of system-level optimization without compromising the quality of the solution obtained.



(a) Power optimization rationale with surplus time slack for a single module.



(b) Power optimization rationale for multiple modules on a single path.



(c) Power optimization rationale for multiple modules on multiple paths.

Figure 4. Power optimization rationale.

1.2 Key Contributions and Research Scope

The goal of this dissertation is power minimization without performance loss for a given target system and a given a range of applications. The proposed method in this dissertation is vertically integrated power optimization across instruction, architecture, and gate-level boundaries. The main contributions of this dissertation are categorized as follows:

- i) Hierarchical delay-driven power-optimization across the hierarchical design boundaries from software level to gate level
- ii) Novel slack distribution by Energy-Delay Ratio (EDR) paradigm
- iii) Ultra-Deep-Sub-Micron (UDSM) awareness during gate-level power optimization

Table 1 shows the research scope of this dissertation according to the full *CR* (computing reviews) classification scheme from ACM.

Table 1. Research scope.

Categories and Subject Description	General Terms	Key Words and Phrases
D.2.10 [Software] Software Engineering: Methodologies C.5.4 [Computer Systems Organization] Computer System Implementation: VLSI Systems B.7.2 [Hardware] Integrated Circuits: Design Aids-simulation	Algorithms, Design, Performance	Low-Power Design, Application-Specific Design, Instruction Scheduling, Time Slack Analysis, Circuit Partitioning and Gate-Level Circuit Optimization

1.3 Overview of the dissertation

Our proposed vertically integrated framework consists of a couple of algorithms at software level, module level, and gate level.

At software level, we minimize total activity of a target system at system level by using our proposed instruction scheduling mechanism (approach I in Figure 5). We solve the instruction scheduling and reordering problems for minimum activity as a Precedence Constrained Hamiltonian Path Problem for directed acyclic graphs (DAGs) and the Traveling Salesman Problem (TSP), both of which are NP-Hard. We propose an efficient instruction-level optimization algorithm for solving the NP-Hard problem. Minimum spanning tree (MST) and simulated annealing (SA) mechanisms are used for the optimization. In addition, a validation method for the optimization is also presented.

For delay assignment at architectural module level, we, first, partition the whole system into functional building blocks and perform activity profiling for the building blocks by using the activity-minimized input vectors from the previous software-level and Monte Carlo simulation. At module level, we hierarchically optimize the time slack in order to minimize power by using the proposed net-based algorithm (approach II in Figure 5) and the path-based algorithm (approach III in Figure 5). To handle the complexity problem of such a hierarchically integrated power-optimal delay assignment, a very simple optimization metric is proposed for the module-level delay assignment. This metric seeks to allocate delays to hardware modules in such a way that the delay of the module is proportional to the energy consumed by the module. We use the minimized

functional-level activity (from software-level analysis) as an input vector for estimating power/energy of each module.

Finally, at gate level, we optimize the technology parameters (approach IV in Figure 5) within the delay constraints from the module-level delay assignment scheme. We implement an efficient approach to minimize total power (switching, short-circuit, and leakage power) tremendously without performance loss for ultra-low power CMOS circuits in nanometer technologies. We present a framework for combining supply/threshold voltage scaling, gate sizing, and interconnect scaling techniques for power optimization, including UDSM effects and back end (post-layout-based) design techniques.

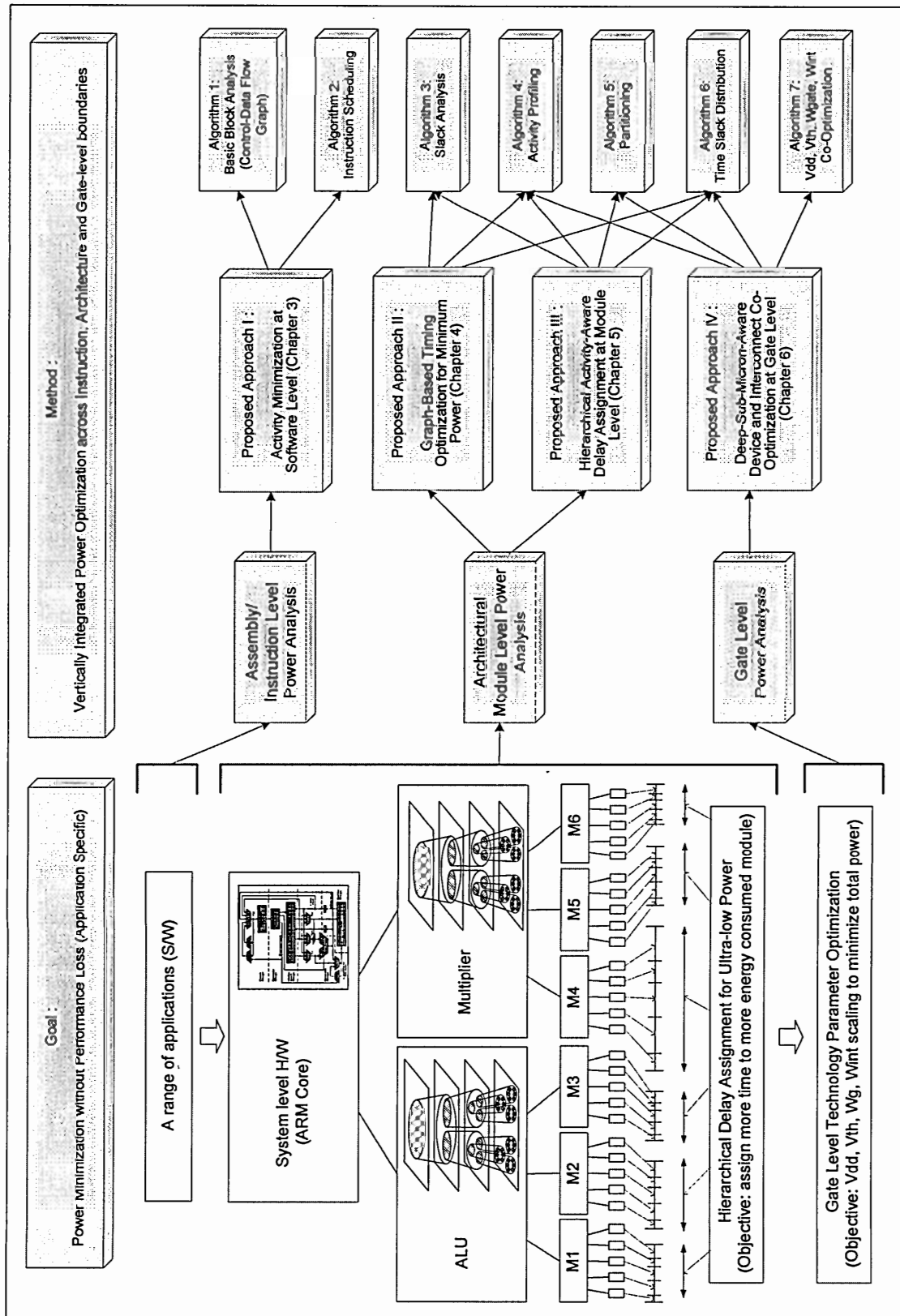


Figure 5. Overview of the proposed approaches.

The remainder of this dissertation is described as follows. The second chapter discusses technology trends, power sources, and previous low-power techniques at each design hierarchy and describes the proposed new design flow. The third chapter illustrates the instruction scheduling algorithm for minimizing total switching activity at software level. For the architectural module-level approaches, the fourth chapter illustrates a net-based time-slack distribution algorithm and the fifth chapter presents a path-based hierarchical delay-assignment scheme that minimizes power consumption. In Chapter 6, we present UDSM-ware gate-level power optimization algorithm for ultra-low-power CMOS VLSI design. Finally, Chapter 7 summarizes the work of this thesis and suggests possible future research directions.

CHAPTER 2

PRIOR WORK AND PROPOSED NEW DESIGN FLOW

The semiconductor technology has continuously improved and has led to ever smaller dimensions of transistors, higher packaging density, faster circuits, and lower power dissipation. Overall technology characteristics of the 2001 edition of the *International Technology Roadmap for Semiconductors (ITRS)* are summarized in Table 2 [2-1]. The overall roadmap technology characteristics tables provide a consolidated summary of the key technology metrics. Historically, in the 1970's the total number of transistors on a chip doubled every 12 months, this trend continued until the 1980's where the trend slowed down, the total number of transistors doubled every 24 months [2-2,2-3]. Wafer sizes continue to grow and the die size has increased with wafer size. Let us assume that performance demand, and consequently the frequency trend continues. Every two years, technology feature size reduces by 50%, electrical nodes in a given area increases by 2x, die size grows by 14%, supply voltage reduces by 15%, and frequency increases by 2x. The primary problem is that in the case of low-power technology, there

is no equivalent of Moore's Law that forecasts a doubling of the complexity of microelectronic chips every 18 months, and Gilder's Law, which theorizes a fourfold increase of the Internet traffic every year. In contrast, energy saving techniques have improved very slowly and only a 20% improvement in capacity will be expected over the next 10 years [2-8]. These trends for the last 6 years are shown in Figure 6.

Table 2. Progress of technology and change of applications [2-1].

Year	1995	2001	2004	2007	2010	2013	2016
LSI technology							
technology node (μm)	0.35	0.13	0.09	0.065	0.045	0.032	0.022
number of transistors	5 million	193 million	386 million	773 million	1546 million	3092 million	6184 million
power supply voltage (V)	3.3	1.1	1.0	0.7	0.6	0.5	0.4
operating frequency (Hz)	300M	1.684G	3.990G	6.739G	11.511G	19.347G	28.751G
power dissipation (W)	50	130	160	190	218	251	288
cell area (μm^2)	-	0.130	0.049	0.024	0.012	0.004	0.002
Application							
computer	PC	Portable		Pervasive/Ubiquitous			
communications	LAN	PAN		Computing			

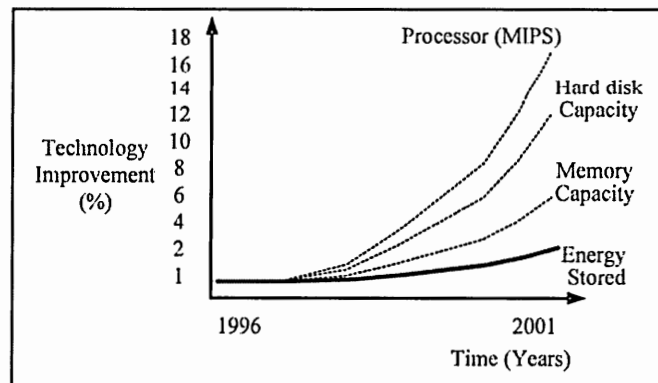


Figure 6. Improvement in technology [2-8].

2.1 Sources of Power Dissipation

Four main sources of current contribute to power dissipation in digital CMOS circuits as shown in Figure 7. These are summarized in Equation (2-1) and discussed below.

$$P_{total} = prob_{TR} \cdot C_L \cdot V_{SS} \cdot V_{dd} \cdot f_{clk} + I_{SC} \cdot V_{dd} + I_{leakage} \cdot V_{dd} + I_{static} \cdot V_{dd} \quad (2-1)$$

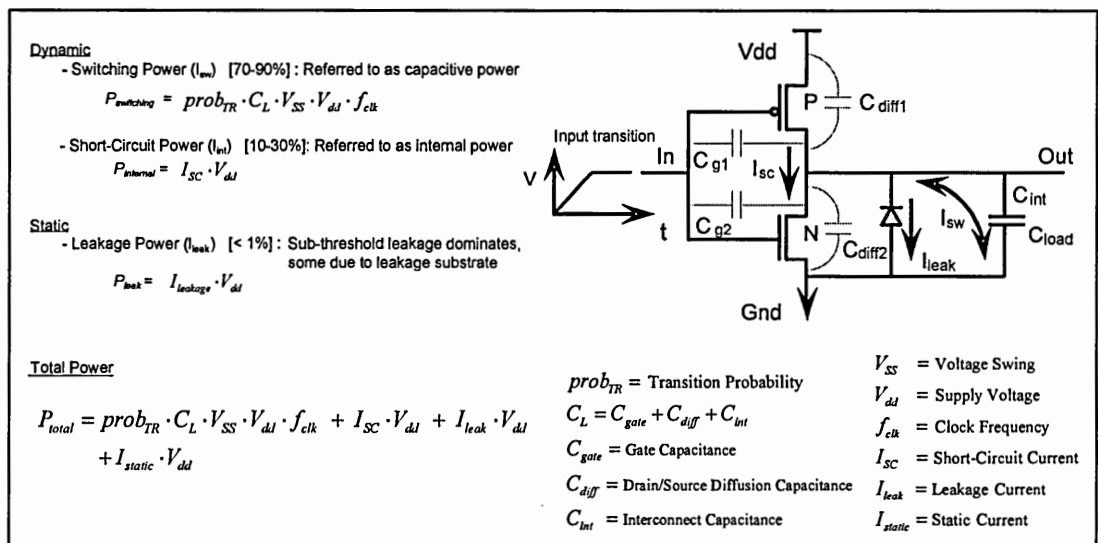


Figure 7. Sources of power dissipation.

- Switching Power:** The first term in the expression for total power in Figure 7 represents the switching component of power, where C_L is the load capacitance, f_{clk} is the clock frequency, and $prob_{TR}$ is the node transition activity factor. The dominant source of power dissipation is the current that flows during charging and discharging the capacitive load that the gate is driving. In most cases, the voltage swing, V_{SS} , is the same as the supply voltage, V_{dd} ; however, some logic gates, such as in single-transistor pass-transistor implementations, the voltage swing on some internal nodes

may be slightly less [2-2]. Sometimes, power can be saved by reducing the voltage swing. This method is particularly useful for high capacitance nodes such as long buses, I/Os etc. [2-3]. Reducing the switching power dissipation has been the subject of extensive study. Most of these methodologies aim at lowering one or more of V_{dd} , $prob_{TR}$ or C_L .

2. **Short-circuit Power:** The second term is due to the direct-path short-circuit current, I_{SC} , which arises when both the NMOS and PMOS transistors are simultaneously active, conducting current directly from supply to ground [2-2]. This current flows when the input waveform rises/falls at a slower rate than the output. Thus, for a short time (when $V_{thn} \leq V_{in} \leq V_{dd} - |V_{thp}|$) both the nFET and the pFET driven by the input remain on, and as a result there may exist a direct current path from V_{dd} to ground. Models for this dissipation component were developed in [2-15]. It has been found that, under typical loads, the short-circuit dissipation is a small fraction ($\approx 10\%$) of the total power dissipation of a CMOS gate.
3. **Leakage Power:** The third term is due to the leakage current, I_{leak} . There are two types of leakage current: (i) reverse-bias diode leakage, which flows in the p-n junctions formed at the substrate-diffusion interface, and (ii) sub-threshold leakage, which occurs when the MOSFET is turned off, but there is still some current flowing because of the inversion charges that exist at gate voltages below the threshold voltage. Both of these currents are determined predominantly by process technology. Leakage currents are usually small and do not pose a major problem in most conventional ICs. However, because of the exponential increase in sub-threshold

currents with reduced threshold voltages, this component becomes significant in low-voltage devices.

4. **Static Power:** Finally, static current, I_{static} , arises from circuits that have a constant source of current between the power supplies such as bias circuitry, pseudo-NMOS logic families, etc. This source of power dissipation, also known as standby dissipation, is due to the DC current that is continuously drawn from V_{dd} to ground. This was a significant contribution in earlier nMOS design styles. In CMOS circuits, however, standby dissipation is not so important, as in steady state there is no direct connection between V_{dd} and ground.

2.2 Summary of Low-Power Techniques

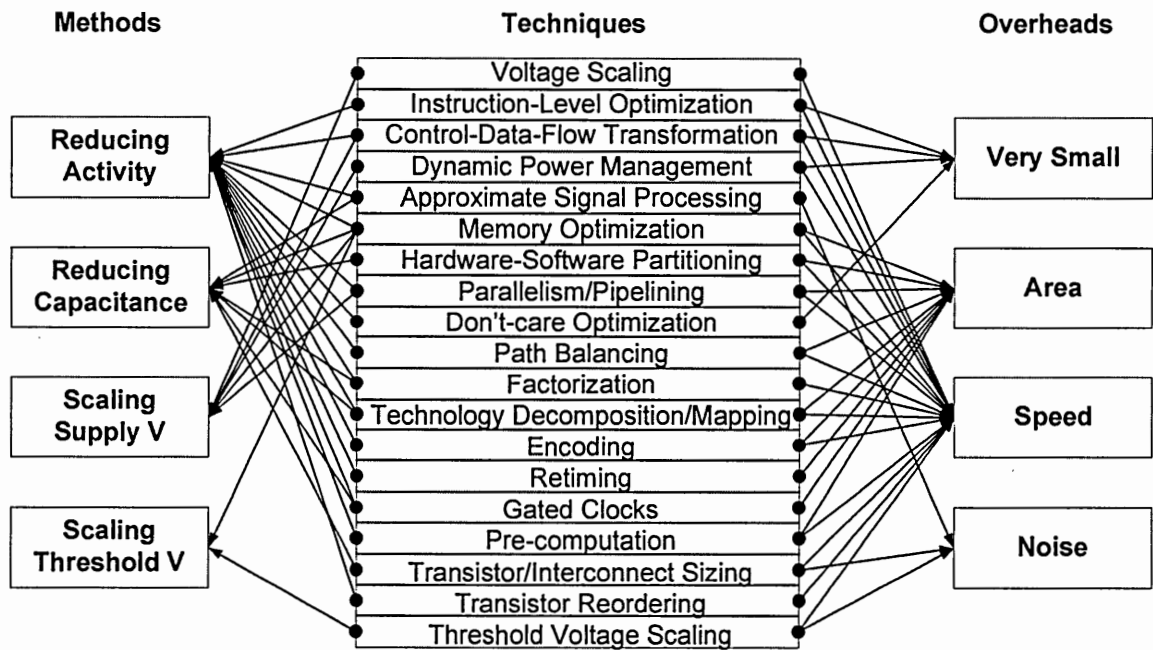
Table 2.3.1 shows a summary of existing low-power design techniques for each level of design hierarchy and for each digital system component.

Table 3. Low-power techniques in design hierarchical view.

Design Hierarchy	Low-Power Techniques							
CIRCUIT /DEVICE	Power Metric : $P_{total} = prob_{TR} \cdot C_L \cdot V_{SS} \cdot V_{dd} \cdot f_{clk} + I_{SC} \cdot V_{dd} + I_{leak} \cdot V_{dd} + I_{static} \cdot V_{dd}$							
	Power Parameters	System Component						
		General	Clock	Bus	Data Path	Random logic	Memory	I/O
	Prob _{TR}		* Gated clock [85]	Glitch Suppress * 3-state-buffer activated after data [122]	* latch insertion to deskew data-in [66]	CAD * permutation of series-connected transistor order [45]		
	C _L	* device scaling [158]	* floorplan to reduce wire length [158] * F/F sizing [48] * C stacking Charge Recycling [159]	* exclusive bus [78] * C stacking [160]	Tr. Reduction * pass-transistor (CPL [2], SRPL [161], SAPL [162])	* tr resizing [35]	* hierarchy [101]	* MCM [166] * area pad [166]
	V _{SS}	Small Signal	* ½ swing [159]	* ½ swing [160]	* pass-transistor (SAPL [162])		* reduced swing WL, BL	* reduced swing I/O (GTL [163])
	V _{dd}	Low V _{dd} * DC-DC converter [164] * 0.25 V QuadRail [165]						
	f _{clk}				* parallelism [2]			* phase modulation [168]
	I _{SC}	Careful Design * design verification by CAD						
	I _{DC}						Cut Current * latch S/A [167]	* dynamic termination [169]
I _{leak}	* control V _{th} fluctuation by SSB [52]				Sleep Mode * 2type V _{th} (MT-CMOS [51])	Switched source-impedance [170]		

Design Hierarchy	Low-Power Techniques							
LOGIC	Power Metric : $P = \alpha \cdot C_{phy} \cdot V_{dd}^2 \cdot f_{clk} = C_{eff} \cdot V_{dd}^2 \cdot f_{clk}$							
	Power Parameters	System Component						
		General	Clock	Bus	Data Path	Random logic	Memory	I/O
	α	* state machine encoding [75]	* clock gating [82]	Signal Control * signal gating [160]	* signal propagation blocking [160]	* Retiming [79] Gate Reorganization * multi-level network optimization [4] * technology decomposition / mapping [70]		
	C_{phy}			* Bus encoding [78]	* path balancing [66]			
	V_{dd}					Multi-Vdd * behavioral synthesis [5]		
f_{clk}								
ARCHI- TECTURE	Power Metric : $P = \alpha \cdot C_{phy} \cdot V_{dd}^2 \cdot f_{clk} = C_{eff} \cdot V_{dd}^2 \cdot f_{clk}$							
	Power Parameters	System Component						
		General	Clock	Bus	Data Path	Random logic	Memory	I/O
	α	* reconfigurable computing [172]		* Bus encoding [174]	* instruction set design [175]		* hierarchical word line [171]	
	C_{phy}			* Bus splicing			* selective precharge [176]	
	V_{dd}	* m-Vdd by parallelism / pipeline [122]					* reduction voltage swing on bit lines [177]	
f_{clk}		* clock generation / distribution [173]						
SOFT WARE	Power Metric : $P = \alpha \cdot C_{phy} \cdot V_{dd}^2 \cdot f_{clk} = C_{eff} \cdot V_{dd}^2 \cdot f_{clk}$							
	Power Parameters	System Component						
		General	Clock	Bus	Data Path	Random logic	Memory	I/O
	α	* algorithm transformation [134] Power management		* bus encoding [174]	Compiler driven * instruction scheduling [132] * register allocation / relabelling [128] * loop unrolling [131]		* minimize mem access [103]	
	C_{phy}							
	V_{dd}	* task scheduling [142]	Compiler driven * Vdd scaling [179]					
f_{clk}		* frequency scaling [146,179]		* software pipelining [178]				

Figure 8 shows typical low-power techniques according to the main approach and the overhead. The methods of the low power techniques are categorized as reducing activities, reducing capacitances, supply voltage scaling, and threshold voltage scaling. The overheads of the low-power techniques can be area, speed, or noise.



References: Voltage Scaling [2-17~2-26], Instruction-Level Optimization [2-124~2-132], CDF Transformation [2-133~2-141], Dynamic Power Management [2-142~2-154], Approximate Signal Processing [2-155~2-157], Memory Optimization [2-89~2-114], Hardware-Software Partitioning [2-115~2-120], Parallelism/Pipelining [2-121~2-123], Don't-care Optimization [2-62~2-65], Path Balancing [2-66,2-67], Factorization [2-68,2-69], Technology Decomposition/Mapping [2-70~2-74], Encoding [2-75~2-78], Retiming [2-79~2-81], Gated Clocks [2-82~2-85], Pre-computation [2-86~2-88], Transistor/Interconnect Sizing [2-27~2-39], Transistor Reordering [2-40~2-45], Threshold Voltage Scaling [2-46~2-61]

Figure 8. Low-power techniques in methodology view.

2.3 Energy and Delay Tradeoffs

In general, there are three ways to save power dissipation while maintaining operation frequency by utilizing surplus time slack: i) employing multiple supply voltage (V_{dd}) to lower supply voltage, ii) employing multiple threshold voltage (V_{th}) to reduce leakage current, and iii) employing multiple device width (W_{gate}) to reduce circuit capacitance. Figure 9 presents the fundamental characteristics of those three device parameters (V_{dd} , V_{th} , W_{gate}) for power and delay tradeoffs [2-4]. Figure 9(a) shows the V_{dd}/V_{th} and (delay \times energy) tradeoffs. It shows that the supply voltage should be larger than four times of the threshold voltage if the delay is not to increase excessively. Figure 9(b) shows the Device Width and (delay \times energy) tradeoffs. It is shown that the delay decreases with increased device width but the delay-energy product is minimized when the devices contribute half of the total load capacitance. The technology parameters trade-offs are summarized in Figure 9(c). In this thesis, we try to optimize the non-linear parameters of those tradeoffs efficiently to minimize the total power.

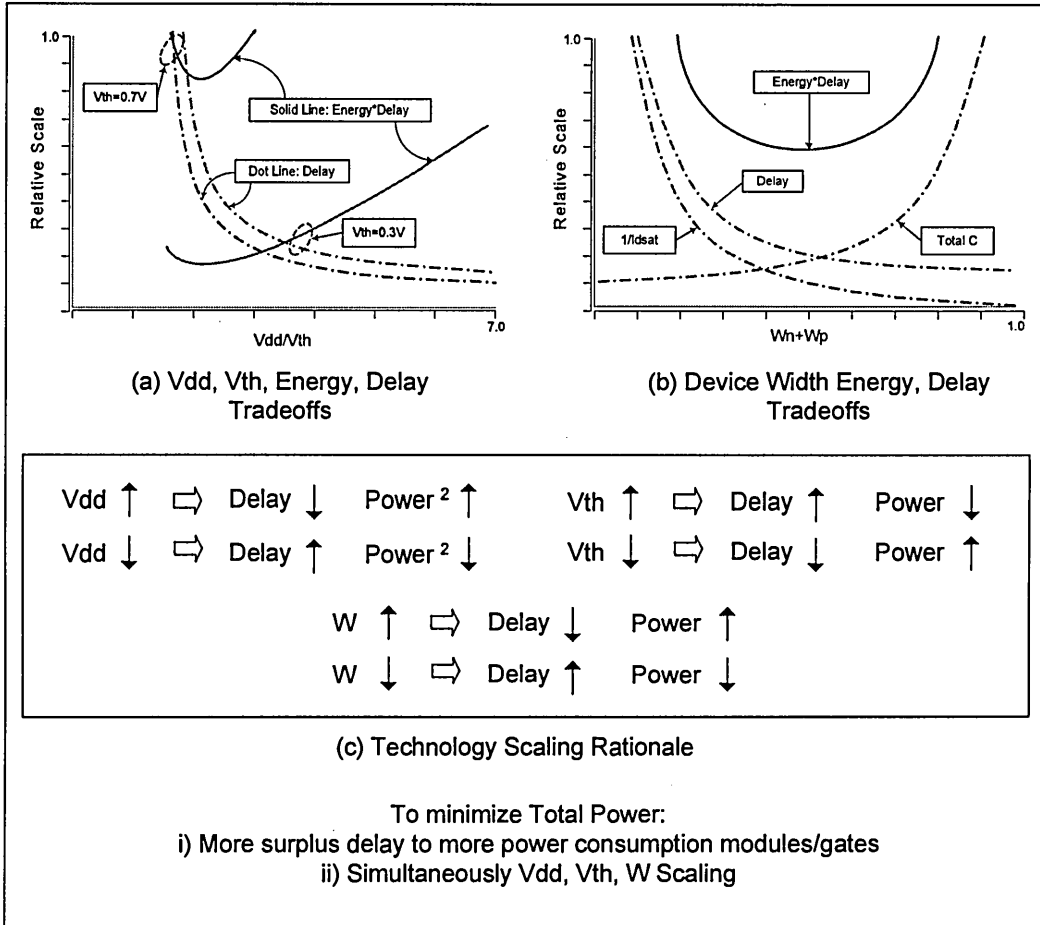


Figure 9. Energy and delay tradeoffs for power optimization.

2.4 Previous Work

2.4.1 INSTRUCTION SCHEDULING

Recently, new research directions in reducing power consumptions for embedded systems have begun to address the issues of arranging software at the instruction level to control power dissipation [2-124~2-132]. The energy consumed by a processor depends

on the previous state of the system and the current inputs. Therefore, it is clear that proper instruction choice and ordering should be considered for low-power systems.

The focus of instruction-level power optimization is closely related to hardware processor systems because these methods are based on a processor power model, where each instruction (pair of instructions) has a power cost. Power consumption per instruction and for a sequence of instructions can be determined accurately for different applications [2-124,2-125,2-132]. After cost analysis, instruction-level scheduling/optimization is performed at compile time. These optimizations consist of instruction set design, instruction packing/ordering, register re-labeling, etc. Some processor architectures support dedicated low-power instruction sets for the purpose of facilitating instruction-level power optimization of the ARM architecture or the Xscale processor [2-129~2-131].

There has been previous research on instruction-level scheduling to reduce total power consumption. In [2-158], a technique called cold scheduling was combined with Gray code addressing to reduce the switching activity in the control path of high-performance processors. The cold scheduling algorithm works much the same way as traditional list scheduling with the exception that it schedules instructions based on a modified priority scheme. Another scheduling technique for reducing power consumption is introduced in [2-124,2-125]. The goal here is to reschedule code such that instructions are more judiciously chosen as opposed to actually reordering instructions to reduce power. In order to achieve the goal of low-power scheduling, the authors implement a methodology for determining the energy consumption of a single instruction in which

they measure the current through the processor when that instruction is executed, using hardware. In [2-159], the authors introduce a redundant search space reduction scheme called BARS. Basically, BARS examines all possible schedules of a DAG, and finds the schedule with the lowest cost. The BARS algorithm employs two techniques for runtime efficiency. One technique avoids potentially redundant search, and another technique limits the number of sub-trees to be searched. Table 4 shows previous results for instruction-level power optimization.

Table 4. Previous results for instruction-level power optimization.

Techniques	Test Circuits/Benchmarks	Power Savings (% _x)
Evaluating power cost of software and Instruction level analysis [2-124]	486DX2 based system	40 %
Optimization heuristics for instruction scheduling [2-132]	Algorithm applications	29 %

2.4.2 TIMING ANALYSIS

A key principle of low power design maintaining speed is the surplus delay (slack budget) distribution among the circuit modules. The first use of the slack distribution approach is the popular zero-slack algorithm (ZSA) [2-160]. The ZSA is a greedy algorithm that assigns slack budgets to nets on long paths for VLSI layout design. It ensures that after the assignment, the net slack budget is maximal, which means that no more slack budget could be assigned to any of the nets without violating the path constraints. Most other slack-distribution algorithms are pruned versions of the ZSA [2-161]. In [1-162], maximum-independence-set algorithm (MISA) shows that the ZSA is far

from the optimal solution in terms of total slack budget and introduces a way to increase the total slack budget only if fan-out numbers of modules in a target circuit are quite large. The ZSA and its off-springs are only for improving delay performance in layout design hierarchy. Therefore, we need a low-power version of time slack distribution algorithm.

2.4.3 SUPPLY VOLTAGE SCALING

Voltage scaling techniques for low power have been investigated for almost all levels of the design hierarchy, from the system level to the device level, as a result of the quadratic effect on the switching power consumption. However, as the supply voltage becomes lower, the circuit delay increases and the performance degrades. There are three ways to maintain performance: i) reduce threshold voltage to improve circuit speed, ii) introduce parallelism into the architecture while using slower device speeds, iii) use multiple supply voltages and choose the lowest supply voltage for different circuit components that satisfies the speed requirements. The power and performance relationship of the first approach is shown in Figure 10. The second approach is discussed in [2-2] in detail. The idea here is to utilize the increasing transistor density to provide additional circuits to parallelize the computation and trade-off silicon area for power consumption. It should be noted that chip sizes are still increasing even though transistor density is increasing by 60% per year [2-3]. It should also be pointed out that there are limits on the degree of parallelism that may be allowable for a given architecture. The problem with the third approach is the need for level conversion every time across

components with different supply voltages. Table 5 presents results for previous techniques of the supply voltage scaling.

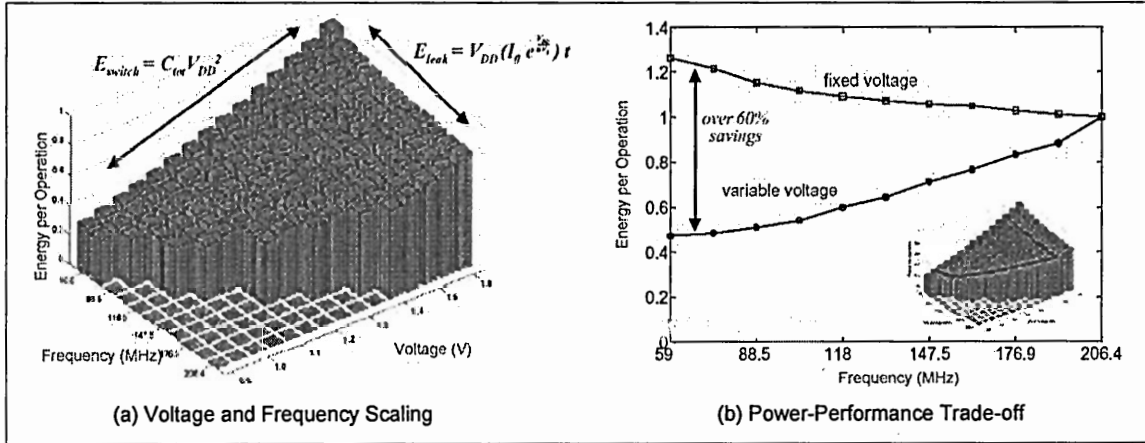


Figure 10. Power-performance tradeoffs for supply voltage scaling [2-26].

Table 5. Previous results for supply voltage scaling.

Techniques	Test Circuits/Benchmarks	Power Savings (%),X
Structuring synthesis, placement, and routing with dual-Vdd [2-24]	Mpact media processor chip	47 %
Timing slack exploration at gate level with dual-Vdd [2-17]	SIS package circuits	19.6 %
Critical Path and Delay Balancing at gate level with dual-Vdd [2-22]	ISCAS 85	25 %
Dynamic programming in pipelined/non-pipelined data paths with four Vdd [2-21]	Test DFG, DSP Filters	43.74 %
multiple supply voltages at microarchitectural level by exploiting the difference in latencies of different pipeline stages [2-25]	SPEC Benchmarks	27 %

2.4.4 THRESHOLD VOLTAGE SCALING

With the scaling of supply voltage, transistor threshold voltage needs to be scaled properly to offset the undesired speed loss. Device threshold voltage is one of the main

determinants of circuit speed at low voltages in CMOS circuits. If supply voltage is greater than four times the threshold voltage, conventional CMOS static circuits are relatively insensitive to supply voltage variation with respect to delay [2-3]. Reducing the threshold voltage improves low-voltage performance with respect to power. Unfortunately, such low threshold voltage design practice not only exponentially increases the leakage power but also deteriorates the noise immunity. Furthermore, given the trend that leakage power increases by a factor of 5X with each technology generation, it will become a significant portion of the total power in future ICs [2-1,2-3,2-4,2-6]. The general mechanism for threshold voltage scaling is that for the active mode of operation, a low threshold voltage is preferred because of the need for higher performance, and for the standby mode, high threshold voltage is useful for reducing leakage power. Different threshold voltages can be obtained by changing the substrate and source bias and by controlling the back gate of double-gate silicon on insulator (SOI) devices [2-6]. Some techniques in the literature are: i) SATS (self adjusting threshold voltage scheme) [2-52]; ii) MTCMOS (multi-threshold voltage CMOS) [2-51]; iii) DTMOS (dynamic threshold voltage MOSFET) [2-53]; and iv) DGDT-SOI (double gate dynamic threshold control SOI) [2-54]. In general, threshold voltage is a function of a number of parameters, including the following: i) gate conductor, ii) gate insulation material, iii) gate insulator thickness-channel doping, iv) impurities at the silicon-insulator interface, and v) voltage between the source and the substrate. Table 6 shows previous results for the threshold voltage scaling.

Table 6. Previous results for threshold voltage scaling.

Techniques	Test Circuits/Benchmarks	Power Savings (% _x)
Transistor level time slack exploration and Mixed-V _{th} static CMOS design technique [2-50]	32-adder, ISCAS 85	80 %
Multiple V _{th} in 1V, 0.5 micron technology [2-51]	16-b fixed-point DSP	10 x
Optimization V _{dd} /V _{th} corresponding to min. energy*delay [2-58]	0.18 micron dual-V _{th} MOSFET	40 %
Dual-V _{th} with criticality exploration [2-59]	ISCAS	50 %
High level synthesis based on the dual-V _{th} [2-25]	GCD, Communication protocols	59 %

2.4.5 GATE SIZING

Transistor and gate sizing affects for dynamic and leakage power reduction and delay. A large gate is required to drive a large load capacitance with acceptable delay but it requires more power. The basic rule is to use the smallest transistors or gates that satisfy the delay constraints. To reduce dynamic power, the gates that toggle with higher frequency should be made smaller. An interesting problem occurs when the sizing goal is to reduce leakage power of a circuit. The leakage current of a transistor increases with decreasing threshold voltage and channel length. In general, a lower threshold or shorter channel transistor can provide more saturation current and thus offers a faster transistor. This presents a tradeoff between leakage power and delay. There have been a number of optimization algorithms for gate sizing for dozens of years [2-27~2-39].

Transistor sizing in a combinational gate circuit can have significant impact on circuit delay and power dissipation. If the transistors in a given gate are increased in size, the delay of the gate decreases. However, the power dissipated by the gate increases. Further, the delay of the fan-in gates increases because of increased load capacitance.

Given a delay constraint, finding the appropriate sizes of transistors that minimize power dissipation is a computationally difficult problem. A typical approach to the problem is to compute the slack at each gate in the circuit, where the slack of a gate corresponds to how much the gate can be slowed down without affecting the critical delay of the circuit. Sub-circuits with slacks greater than zero are processed and the sizes of the transistors reduced until the slack becomes zero or the transistors are all minimum size. Table 7 shows previous results for this technique.

Table 7. Previous results for gate sizing.

Techniques	Test Circuits/Benchmarks	Power Savings (% _x)
Downsizing MOSFETs inside a cell by redundancy cell-based circuits elimination [2-27]	SICAS 85, LGSynth 93	65 %
TILOS heuristic [2-30]	16-adder, RISC logic, Video RAMDAC	40 % - 50 %
Analysis parasitic capacitances and velocity saturation of micron technology [2-39]	Asynchronous prescaler (four F/Fs) in 1.0 micron CMOS	30 % - 50 %

2.4.6 INTERCONNECT SCALING

The interconnect delay and power have become dominating factor in determining the circuit performance in ultra-deep-submicron designs because of mainly two reasons: i) increased routing densities that have led to shrinking interconnect pitches and ii) aspect ratios that attempt to keep the resistance of these narrower interconnects constant. Both have resulted in an increase in the capacitance per unit length, particularly due to interlayer coupling capacitance. Interconnect scaling approaches have been introduced for the optimal interconnect structure of each net in terms of interconnect topology, wire

width (W_{int}) and spacing, buffer locations and sizes to meet low-power demand, speed limit, and signal reliability requirements [2-163~2-165].

2.5 The New Design Flow

The key step of the proposed vertically-integrated design flow is shown in Figure 11. Starting at the top of Figure 11, the first high-lighted box represents the use of instruction-level approach to minimize the total activities in a target system. For a given architecture, activity profiling and hierarchical circuit partitioning are performed. Then, beginning with the topmost level of the design hierarchy, delay values are assigned to every module at that level as denoted in the second high-lighted box in Figure 11. The total delay from primary input (PI) to primary output (PO) is given. The problem is to determine the delays of the individual modules so that total power consumption is minimized by optimizing the supply voltage, threshold voltage and device/interconnect sizes of module for the assigned delay values. The procedure is repeated hierarchically. We use the proposed EDR-based heuristic to assign delays to each module. Given the assigned delay for each module, we optimize technology parameters (V_{dd} , V_{th} , W_{gate} , W_{int}) at gate level as presented in the third high-lighted box in Figure 11. At this stage, UDSM effects and total power (switching, short-circuit, and leakage power) are considered and the optimal technology parameters are provided for each module.

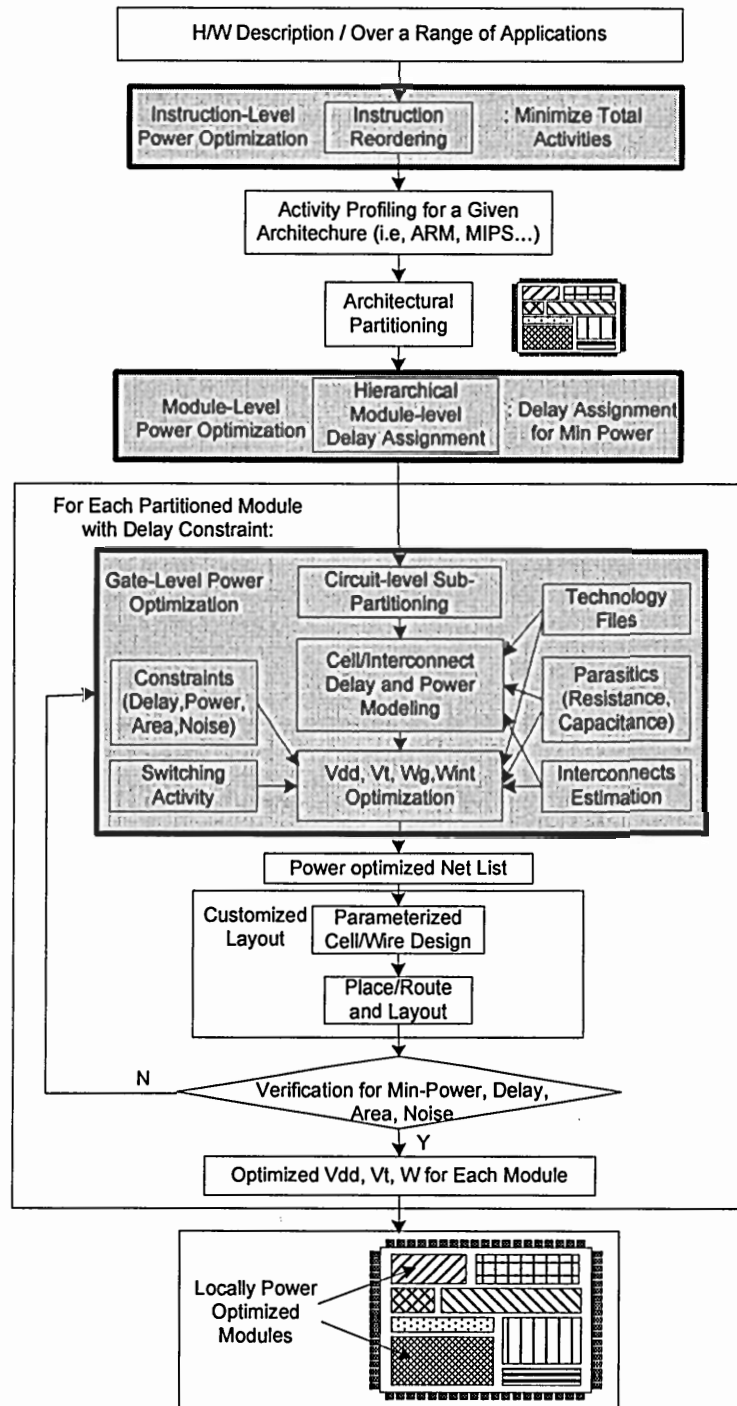


Figure 11. Proposed design flow.

CHAPTER 3

LOW-POWER INSTRUCTION SCHEDULING

3.1 Introduction

In this chapter, we propose an efficient instruction-level optimization algorithm for low power at software level of the design hierarchy. The proposed scheme has the following features: i) the instruction-ordering problem for low power is formulated as a combinatorial graph-search optimization problem, yielding near-optimal results, ii) an efficient heuristic is proposed to solve the problem, iii) cycle-by-cycle RTL-level power simulators are implemented to drive the optimization process, and iv) a methodology for validating the results of the optimization process is presented.

3.2 Problem Description

The general procedure for instruction-level analysis is shown in Figure 12:

- 1) Assembly instructions are assembled from the source program.
- 2) The assembly code is decomposed into Basic Blocks (BB).

3) Control and data dependency graph (CDG) is constructed.

After obtaining the CDG, instruction scheduling is performed for each BB and optimal instruction code for low power is provided.

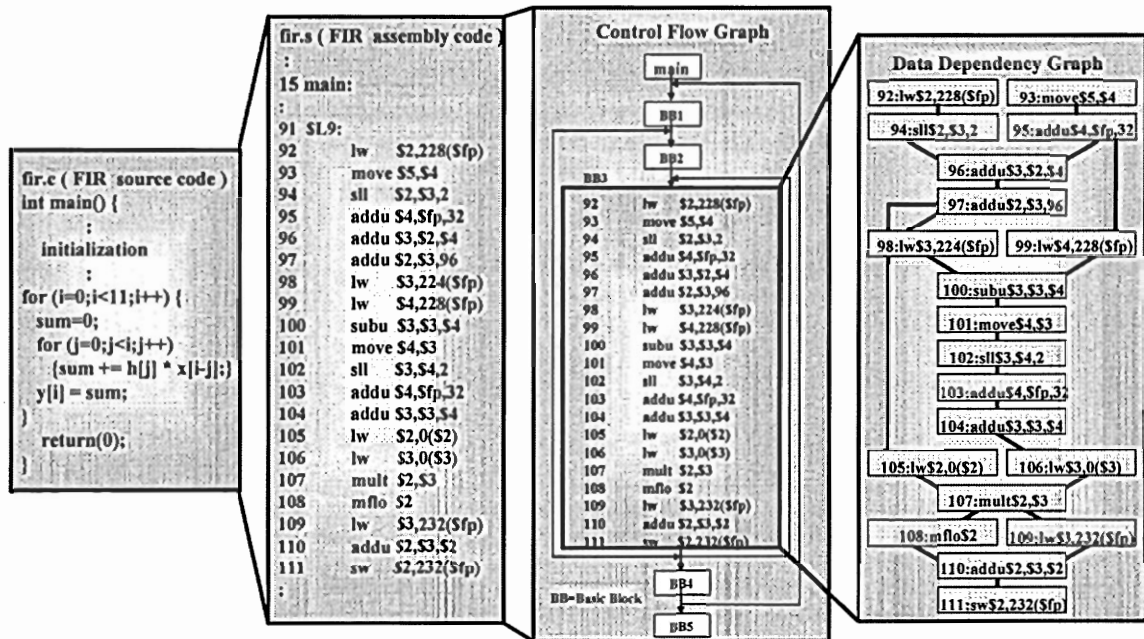


Figure 12. Pre-procedure for instruction scheduling.

It is well known that the choice of instruction sequences is critical to reducing power consumption for embedded systems design. The problem is the complexity of the instruction scheduling. With n instructions, there are $(n-1)!!/2$ possible instruction sequences. For example, for 11 instructions, there are 16,314,400 possible combinations of the instruction sequences. For large numbers of instructions, it is practically impossible to examine all possible instruction sequences to find the optimal sequence that minimizes power consumption. An example of search space for possible instruction sequences is

3.3 Instruction-Level Power Optimization Algorithm

The procedure of the instruction-level optimization methodology is given below:

Step1: Assemble the source code

Step2: Decompose into basic blocks

Step3: Construct control and data dependency graph (CDG) for each basic block

Step4: Generate power dissipation table (PDT) from power simulation

Step5: Build weighted strongly connected graph (SCG) from CDG and PDT

Step6: Solve traveling salesman problem (TSP) problem to find optimal or near optimal solution

Step7: Validate the optimization algorithm

Step8: Generate instruction sequence for low power

Now, we discuss specific steps of the optimization procedure. Figure 14 provides an example of the optimization procedure. Figure 14(a) shows a power dissipation table (PDT) that shows the average power consumed when an instruction in the left most column is followed by an instruction in the topmost row. The control flow and data dependency graph (CDG) of Figure 14(b) shows the inter-dependency of instructions. For example, instruction ④ can be executed only after both instruction ① and ② have both been executed, while instructions ①, ②, and ③ may be executed in any order. The weighted strongly connected graph (SCG) of Figure 14(c) contains all the edges of the CDG in Figure 14(b) and some additional edges. If the order in which any two instructions are executed can be interchanged without violating the constraints of the

CDG, then an additional edge is inserted between the graph nodes (vertices) corresponding to the two instructions. For example, the instructions ④ and ⑤ of the CDG in Figure 14(b) can be interchanged. Hence, in Figure 14(c), there is an edge between the nodes ④ and ⑤. Each edge of the SCG is assigned a weight corresponding to the power consumed by running the pair of instructions repeatedly. It is assumed that the order in which an instruction pair is executed does not matter. Once the SCG is obtained, a minimum spanning tree algorithm (MST) in Figure 14(d) is used to obtain an initial sequence for the optimization. After the MST is performed, local improvement heuristics are conducted by using simulated annealing to obtain an optimal (least cost) Hamiltonian Tour of all the vertices of the SCG as shown in Figure 14(e). This tour gives the optimal sequence of instructions from the viewpoint of power. In this case, the optimal sequence is ③,②,①,④,⑤,⑥.

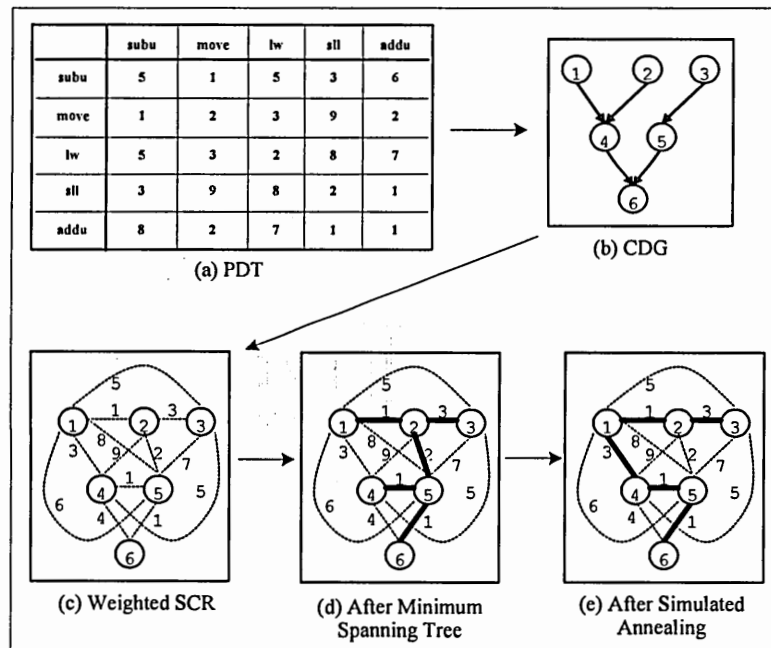


Figure 14. Proposed instruction-level optimization approach.

3.3.1 CDG CONSTRUCTION

First, we review basic graph notations that are used in this research. A graph G consists of a set of nodes V , and a set of edges E , $G = \langle V, E \rangle$. The degree of a node is the number of neighbors adjacent to it. We write $X \rightarrow Y$ when $\langle X, Y \rangle \in E$ in a directed graph. We define $PRED(X)$ as the set of the all predecessors of node X , and $SUCC(X)$ as the set of all successors of X . If $X \rightarrow Y$, then $X \in PRED(Y)$ and $Y \in SUCC(X)$. The *indegree* of a node X is the cardinality of $PRED(X)$ and the *outdegree* of a node X is the cardinality of $SUCC(X)$. The *indegree(X)* is the number of edges of the form $W \rightarrow X$, and the *outdegree(X)* is the number of edges $X \rightarrow Y$. If $W \rightarrow X$, then $W \in INDEG(X)$ and $X \in OUTDEG(W)$. So, the $OUTDEG(W)$ is a sub set of $SUCC(W)$, and the $INDEG(X)$ is the sub set of $PRED(X)$. The notation $X \ast \rightarrow Y$ is to mean that there is a path from X to Y . The control flow and data dependency graph specifies the execution order among the components for given instructions to obey the original program semantics. Figure 15 shows the CDG construction algorithm.

Algorithm 1 Algorithm for CDG Construction

Input: An instruction-level source code

Output: Control and Data flow DAG representation

Definition: **BB** : Basic Block

idn:x : id number (**idn**) for each instruction **x**

src_pre(x) : previous source operands of **x**

des_pre(x) : previous destination operands of **x**

src(x) : current source operands of **x**

des(x) : current destination operand of **x**

step 0: Set identify number (**idn:x**) for each instruction **x**.

step 1: Calculate total BB number and store the start **idn** of each **BB**.

step 2: For each **x** in **BB**, visit **x**, identify **src_pre(x)**, **des_pre(x)**, **src(x)**, and **des(x)**.

step 3: IF there is any **y** in **des_pre(x)** associated with **src(x)**, THEN create an $\langle x,y \rangle \in E$ in $G = \langle V,E \rangle$ and remove **y** from **des_pre(x)**

step 4: IF there is any **y** in **des_pre(x)** associated with **des(x)**, THEN create an $\langle x,y \rangle \in E$ in $G = \langle V,E \rangle$ and remove **y** from **des_pre(x)**

step 5: IF there is any **y** in **src_pre(x)** associated with **des(x)**, THEN create an $\langle x,y \rangle \in E$ in $G = \langle V,E \rangle$ and remove **y** from **src_pre(x)**

step 6: IF there is any **x** in **BB** to be scheduled, THEN go to **step 2**.
ELSE generate CDG graph and return.

Figure 15. CDG construction algorithm.

3.3.2 PDT GENERATION

The power dissipation table describes the power consumption due to the execution of an instruction itself and a pair of instructions. We use a cycle-accurate data path energy model at the RTL level. A cycle-by-cycle instruction-level simulator is used to collect the switching activity at all latches in the data path during execution of the program. The SimpleScalar tool set [3-11], version 3.0 with RTL level power model is

used. Our target machine has a five-stage pipelined data path and the functional units are 32-32bit general registers, I-cache, D-cache, divider, multiplier, ALU, shifter, and control units. Details of generating infinite loops to measure the power dissipation due to an instruction itself and an instruction pair are shown in Figure 16. To avoid any corruption due to loop overhead, we repeat the instruction pair 200,000 times with a *nop* operation.

```

main:
:
lw  $2,228($fp) }
move $5,$4      }
nop             }
lw  $2,228($fp) }
move $5,$4      }
nop             }
lw  $2,228($fp) }
move $5,$4      }
nop             }
... 200,000 times
:

```

Figure 16 PDT generation example.

3.3.3 SCG GENERATION

We define a strongly connected graph that is a set of nodes S such that there is $S_1 \rightarrow S_2$ for any two nodes $S_1, S_2 \in S$. As shown in Figure 14(c), the weighted SCG is a subset of all possible instructions corresponding to the CDG. The SCG generation algorithm is shown in Figure 17.

Algorithm 2 Algorithm for SCR Extraction

Input: CDG stream
Output: SCR DAG representation
Definition: $\text{indegree}(X)$: the number of edges of $W \rightarrow X$
 $\text{outdegree}(X)$: the number of edges of $X \rightarrow Y$
 $\text{INDEG}(X)$: If $W \rightarrow X$, then $W \in \text{INDEG}(X)$
 $\text{OUTDEG}(X)$: If $X \rightarrow Y$, then $Y \in \text{OUTDEG}(X)$
 $\text{SCC}(X)$: strongly connected components of X
 means that the components in $\text{SCC}(X)$
 can be followed by X
 (i.e., $\text{SCC}(X) = [Y, Z]$: Y, Z can be followed by X)

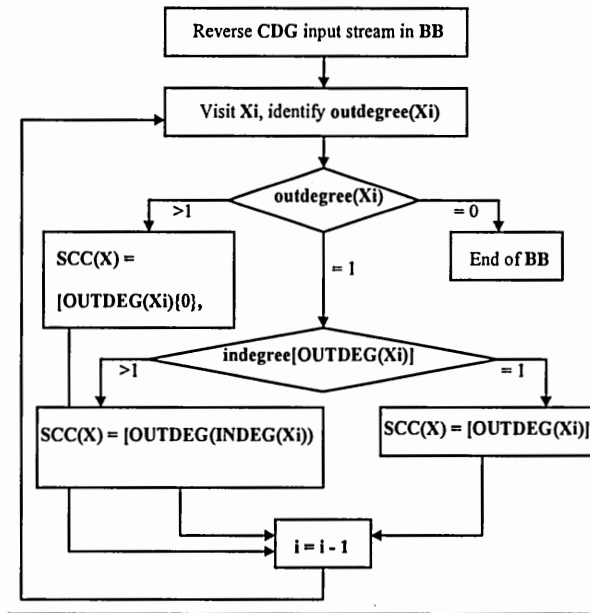


Figure 17. SCG generation algorithm.

3.3.4 TSP IMPLEMENTATION

In the general form of the traveling salesman problem (TSP), we are given a finite set of points V and a cost c_{uv} of travel between each pair $u, v \in V$. A tour is a circuit that passes exactly once through each point in V . The TSP is to find a tour of minimal cost. The TSP can be modeled as a graph problem by considering a complete graph $G = \langle V, E \rangle$, and assigning each edge $u, v \in E$ the cost c_{uv} . A tour is then a circuit in G that meets every node. The tours are sometimes called *Hamiltonian Circuits*. In this research,

as a construction method, we use a minimum-cost spanning tree by using Prim's Algorithm [3-13], and as an improvement method, we use *2-optimal* and *3-optimal* local search heuristics by using simulated annealing. Prim's algorithm is used for computing a minimum spanning tree. It builds upon a single partial minimum spanning tree, at each step adding an edge connecting the vertex nearest to but not already in the current partial minimum spanning tree. If Prim's Algorithm is applied to the graph of Figure 18 (a) with starting vertex *a*, edges are chosen in the order *ab*, *af*, *ac*, *cd*, *dg*, *de* as shown in Figure 18 (b).

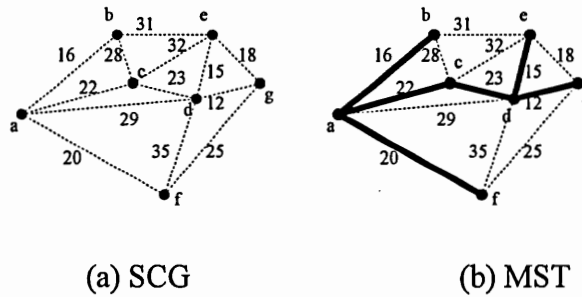


Figure 18 Prim's algorithm example for MST.

The MST solution is used as a starting point for further optimization using heuristics. In this work, we use *2-optimal* and *3-optimal* mechanisms for the improvement heuristic, using simulated annealing with the reasonable cooling parameters as the optimization engine. The *2-optimal* heuristic proceeds by considering each non-adjacent pair of edges of tree *T* in turn. If these edges are deleted, the *T* breaks up into two paths *T*₁ and *T*₂. There is a unique way that these two paths can be recombined to form a new tour *T'*. If $c(T') < c(T)$, then we replace *T* with *T'* and repeat the procedure.

As shown in Figure 19, if $c(T') > c(T)$ for every choice of pairs of nonadjacent edges, then T is *2-optimal* and we terminate the procedure.

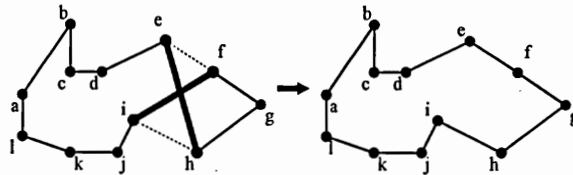


Figure 19 2-interchange local Search heuristics.

The *2-optimal* algorithm can be generalized naturally to a *k-optimal* algorithm, wherein we consider all subsets of the edge-set of a tour of size of k , or size at most k , remove each subset in turn, then see if the resulting paths can be recombined to form a tour of lesser cost. The problem is that the number of subsets grows exponentially with k , and we soon reach a point of diminishing return. For this reason, *k-optimal* for $k > 3$ is rarely used. Figure 20 shows *3-optimal* example. In this research, we use both the *2-optimal* and the *3-optimal* heuristics for optimization.

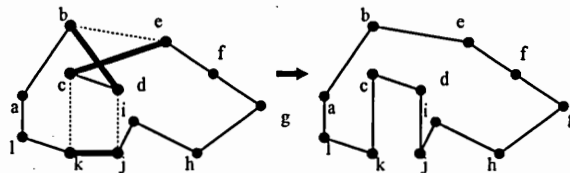


Figure 20 3-interchange local search heuristics.

3.4 Validation of the Optimization

Let us designate the sample space $n=(k-1)!/2$ for k instructions in a BB. Let us define ν as an input vector for an instruction sequence, define random variable $W(\nu)$ to be the power dissipation of the system when ν is applied. If we choose j input vectors $\nu_1, \nu_2, \dots, \nu_j$ randomly and exclusively, the power dissipation obtained in each trial is also an independent random variable (RV) $W_j=W(\nu_j), j=1, \dots, n$. Since the W_j are independent, the set $\{W_1, W_2, \dots, W_j\}$ constitutes a random sample. We define a new RV $X_j=\min(W_1, W_2, \dots, W_j)$, so that X_j is the minimum power dissipation over j trials.

Let us designate the j th ordered observation oX_j and the fraction of the population below this observation as $P_j=F(oX_j)$. Since the value of oX_j varies from sample to sample, P_j is a random variable. Thus, we must consider the distribution of P_j . We will now proceed with the development of the distribution for P_j .

Given an ordered random sample oX_1, oX_2, \dots, oX_n of size n from a population having a cumulative distribution function $F(X)$, where X is continuous, we try to find estimators for $F(oX_1), F(oX_2), \dots, F(oX_n)$. Let us define ${}_n P_j=F(oX_j)$, where ${}_n P_j$ is the probability of failing to find an optimal minimum sequence X_n prior to the j th ordered observation in a sample space n . Then, we have the $F(oX_j)$ probability for $j-1$ outcomes, $\int f(oX_j)dX$ for 1 outcome, and $1- F(oX_j)$ for $n-j$ outcomes. Clearly, the multinomial distribution is applicable in this situation. Recall that the multinomial distribution is given by

$$P(y_1, y_2, \dots, y_k) = \frac{m!}{y_1! y_2! \dots y_k!} (\Theta_1)^{y_1} (\Theta_2)^{y_2} \dots (\Theta_k)^{y_k} \quad (3-1)$$

where, the Θ_i is the probability of obtaining the i th outcome and y_i is the RV representing the number of outcomes of the i th type.

Then, the probability of $j-1$ outcomes with $F({}_oX_j)$, 1 outcome with $f({}_oX_j)dX$, and $n-j$ outcomes with $1-F({}_oX_j)$ is given by

$$\frac{n!}{(j-1)!(n-j)!} [F({}_oX_j)]^{j-1} f({}_oX_j) dX [1-F({}_oX_j)]^{n-j} \quad (3-2)$$

This is precisely the distribution of ${}_oX_j$, the value of the j th ordered observation.

Now we know that

$$dF({}_oX_j) = f({}_oX_j) dX = d_n p_j \quad (3-3)$$

Hence, the probability element becomes

$$g({}_n p_j) d_n p_j = \frac{n!}{(j-1)!(n-j)!} {}_n p_j^{j-1} (1-{}_n p_j)^{n-j} d_n p_j \quad (0 \leq {}_n p_j \leq 1) \quad (3-4)$$

This distribution is commonly termed the rank distribution. Actually the p.d.f. of the RV ${}_n P_j$ is the well-known beta distribution [3-14]. Figure 21 shows the p.d.f. for different values of j for a sample of size $n=10$.

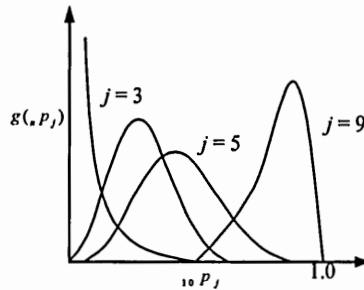


Figure 21 The rank distribution for a sample of size 10.

Recall that the ${}_n P_j$ is the probability of failing to find an optimal minimum sequence X_n prior to the j th ordered observation in a sample space n . Let us define that ε is the error tolerance with a range of $0 \leq \varepsilon \leq 1$ and α is confidence limit with range $0 \leq \alpha \leq 1$.

$$P\{{}_n P_j \leq \varepsilon\} \geq \alpha \quad (3-5)$$

This equation describes the method used to estimate the confidence limit of the optimization with error tolerance ε . For example, in order to get 95% confidence with less than 5% error tolerance for any optimization value, it is estimated as follows:

- 1) Estimate j trial number from the $P\{{}_n P_j \leq 0.05\} \geq 0.95$ in the sample space n
- 2) Generate j input instruction sequences randomly and exclusively
- 3) Find the minimum value among all j trial sequences
- 4) Compare the minimum value (minimum power) and TSP optimized value
- 5) If two values are close, we can say that the TSP optimization has 95% confidence limit with 5% error tolerance.

3.5 Experimental Results

All simulations are done by using the SimpleScalar tool set [3-11], version 3.0 with RTL level power model. The target machine includes a five-stage pipelined data path, 32-32bit general registers, I-cache, D-cache, divider, multiplier, ALU, shifter, and control units. The simulated annealing parameters are tuned for 95% confidence limit and 5% error in Equation (3-5). We assume a fixed clock frequency, fixed supply voltage, and 0.35 micron technology. So, the terms “power” and “energy” can be used interchangeably because the clock frequency is fixed. Moreover, the power is almost proportional to the switching capacitance, due to the fact that over 90% of the total power dissipation is the switching power in well-designed circuits [3-15]. We analyze the power consumption without any schedule and with the proposed optimal scheduling method when running several embedded applications.

Figure 22 shows an optimal instruction sequence example for an FIR filter application. When one basic block is optimized, 8.5 % of total power can be reduced with our scheme. Figure 23 presents power savings in each component of the target system. Table 8 shows that maximum 29% of the total activity can be reduced with our algorithm about 95% confidence limit with 5 % error tolerance.

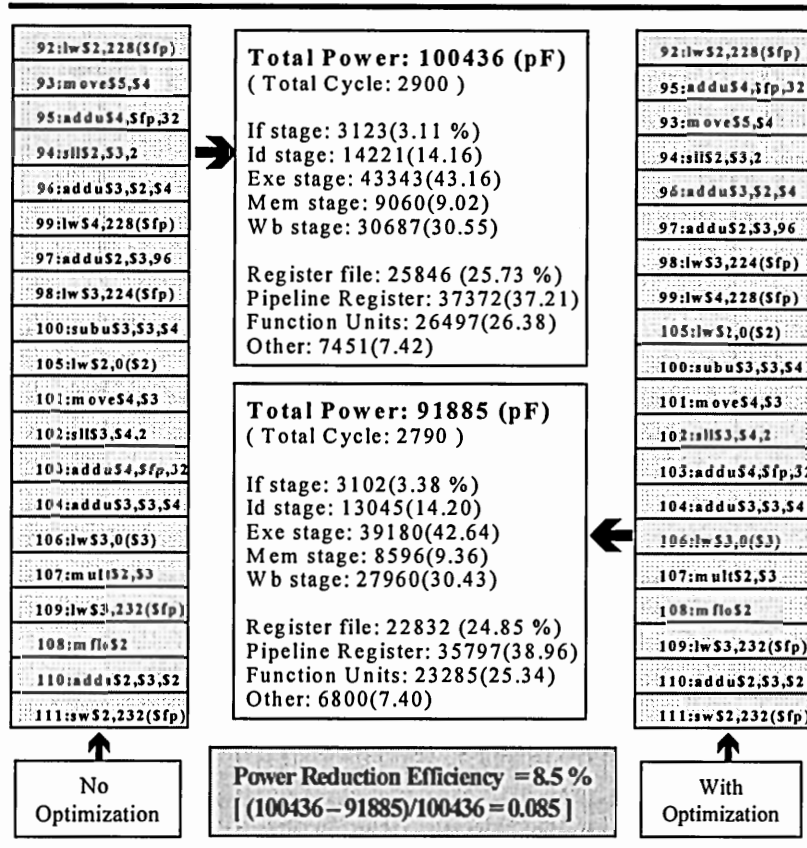
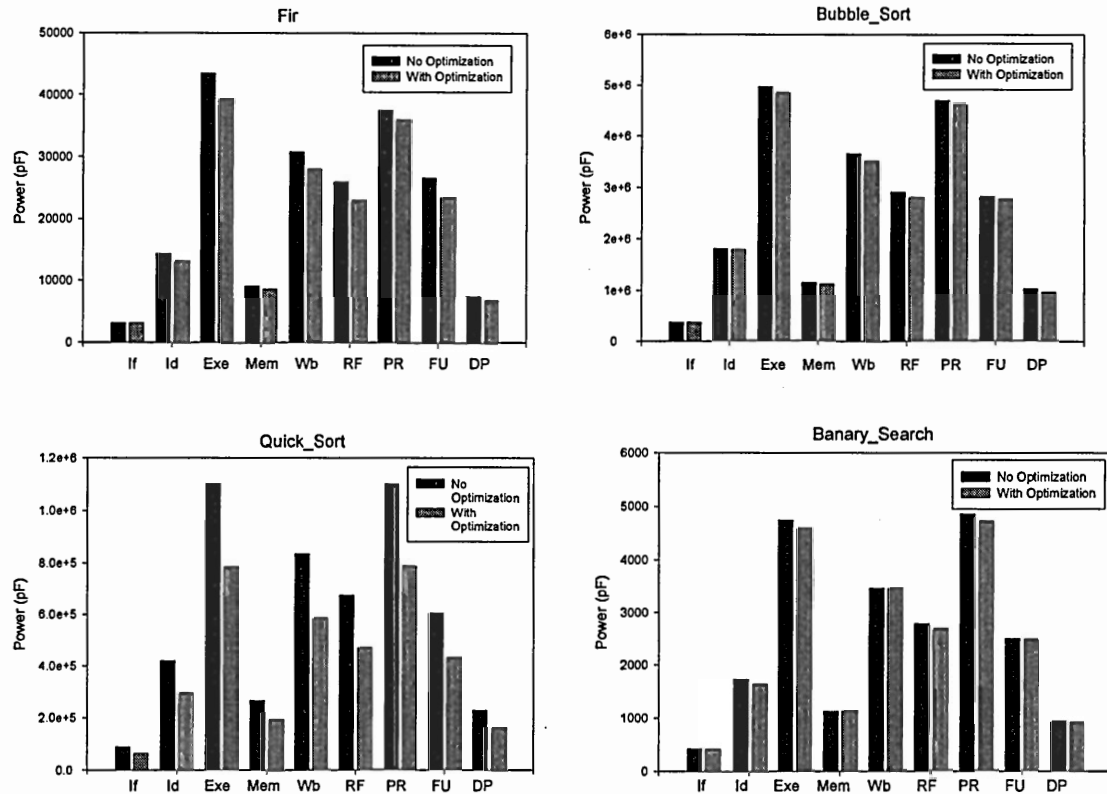


Figure 22. Scheduling example for one basic block of FIR source.



Note: If: Instruction Fetch, Id: Instruction Decode, Exe: Execution, Mem: Memory access, Wb: Wright back, RF: Register file, PR: Pipeline Register, FU: Functional Units, DP: Other Data Paths

Figure 23. Power savings for each component .

Table 8. Total power saving results for some applications.

	Total Power / (Cycles) without Scheduling (pF)	Total Power / (Cycles) with Scheduling (pF)	Confidence Limit / Error Rate (%)	Activity Reduction (%)
FIR Filter	100436.35 / (2900)	91885.50 / (2790)	≈ 95 / ≈ 5	8.52
Bubble Sort (100 Samples)	11948917.22 / (368100)	11627652.60 / (360675)	≈ 95 / ≈ 5	2.68
Quick Sort (100 Samples)	2711445.75 / (85502)	1919845.60 / (61170)	≈ 95 / ≈ 5	29.19
Heap Sort (100 Samples)	3615121.35 / (101136)	3373014.48 / (96329)	≈ 95 / ≈ 5	6.69

3.6 Summary

In this chapter, we proposed an instruction-level optimization algorithm that solves the instruction scheduling and reordering problem for low-power systems design. The proposed algorithm contributes for reducing switching activity (hamming distance) at software level of the proposed design hierarchy, we solve the instruction scheduling and reordering problem as a Precedence Constrained Hamiltonian Path Problem for DAGs and the Traveling Salesman Problem (TSP), both of which are NP-Hard. We proposed an efficient instruction-level optimization algorithm to solve the NP-Hard problem. Minimum spanning tree (MST) and simulated annealing (SA) mechanisms were used for the optimization. We describe the methods for generating the control flow and data dependence graph (CDG), power dissipation table (PDT), and weighted strongly connected graph (SCG) for the instruction-level low-power analysis. In addition, confidence limits with error tolerance were considered for the validation of the optimization. Finally, experimental results that demonstrate the effectiveness and the efficiency of the proposed algorithms were shown.

CHAPTER 4

GRAPH-BASED TIMING ANALYSIS FOR MINIMUM POWER

4.1 Introduction

In general, low-power optimizations that do not compromising performance are dependent on time slack calculation and the surplus slack (slack budget) distribution. The time slack means the difference between the signal required time and the signal arrival time at the primary output of each module. In actual designs, critical paths are small portions of the circuit. In other words, excessive slack remains in the rest of the circuit. Designers have to struggle with the slacks to minimize their power dissipations up to meeting the timing constraints within a required design time. A number of timing-driven approaches have been proposed. They are mainly categorized as: i) weighted net-based algorithms and ii) critical path-based algorithms. In the weighted net (topological depth-based or stage-based) algorithm, timing constraints on the paths are translated into their

timing upper bounds for each net using a time slack distribution approach and the calculated slacks are used for optimizing design criteria such as speed, area, and power in the circuit [4-1~4-6]. In the critical path-based algorithms, path delays are analyzed explicitly and the criticalities on the non-critical paths are used for minimizing power [4-7,4-8]. This research focuses on the problem of the net-based slack distribution for ultra low-power CMOS circuits.

First use of the slack distribution approach is the popular *zero-slack algorithm* (ZSA) [4-3]. The ZSA is a greedy algorithm that assigns slack budgets to nets on long paths for VLSI layout design. It ensures that after the assignment, the net slack budget is maximal, which means that no more slack budget could be assigned to any of the nets without violating the path constraints. Most other slack-distribution algorithms are pruning versions of the ZSA [4-4~4-6]. In [4-6], *maximum-independence-set algorithm* (MISA) shows that the ZSA is far from the optimal solution in terms of total slack budget and introduces a way to increase the total slack budget only if fan-out numbers of modules in a target circuit are quite large. The ZSA and its off-springs are only for improving delay performance in layout design hierarchy. In this paper, we propose a low-power version of ZSA. The proposed PA-ZSA shows that our surplus slack distribution strategy ensures power optimal design without delay performance degradation for CMOS logic circuits.

4.2 Graph-Based Timing Analysis

A CMOS random logic network can be represented as a directed acyclic graph $G=(V,E)$ where each node $v \in V$ represents a logic gate, and a directed edge $e_{ij} \in E$ exists if the output of gate i is an input to gate j . A primary input (**PI**) node has no fan-in (f_i) edges and a primary output (**PO**) has no fan-out (f_o) edges. A combinational circuit operates properly only within the satisfied functional and timing specifications. The timing specification is given as the *arrival time* at each primary input and the *required time* at each primary output.

Let $f_i(v)$ be the i th fanin of node v and let u_i be the node driving $f_i(v)$. Also let $t_{u,v}$ be the propagation time from node u_i to node v and t_v be the delay of node v (see Figure 24).

Definition 1: The *arrival time* of signals at node v is given by the time at which the last (latest) signal arrives at the output of v and is given by

$$arr(v) = \max_{i \in (1, f_i(v))} \{arr(u_i) + t_{u_i, v} + t_v\} \quad (4-1)$$

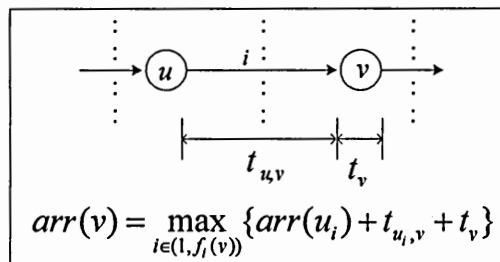


Figure 24. Arrival time.

Let $f_o(u)$ be the set of fan-outs of node u .

Definition 2: The *required time* of signals at node u is given by the time at which the early (earliest) signal to be required at the output of u and is given by

$$req(u) = \min_{i \in (1, f_o(u))} \{req(v_i) - t_{v_i} - t_{v_i, u}\} \quad (4-2)$$

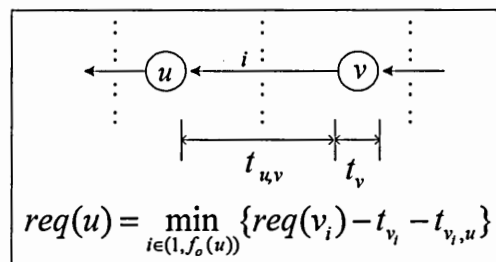


Figure 25. Required time.

Definition 3: The *slack* at node v is the difference between the require time and the arrival time at v .

$$slack(v) = req(v) - arr(v) \quad (4-3)$$

If a circuit initially meets its timing constraints, we have slack time $slack(v) \geq 0$ for each node v .

Definition 4: The node v is defined to be *slack-sensitive* to node u if a change in the delay of node u affects the slack of node v . A slack-sensitive sub-graph (path) of $G(V, E)$ consists of a set of nodes of $G(V, E)$ that are mutually slack sensitive to each other.

The node v is defined to be slack-sensitive to node u if an change in the delay of node u affects the slack of node v . A slack-sensitive sub-graph (path) of $G(V,E)$ consists of a set of nodes of $G(V,E)$ that are mutually slack sensitive to each other.

As an aid to analyzing timing characteristics, we used symbols for the graph-based analysis as shown in Figure 26. The node name and energy consumption (simply activity factor or effective capacitance) is located above the node.

Definition 5: The *effective capacitance* of a module is defined to the sum of the products of the activity factors at each circuit node of the module and the corresponding switched capacitance at that node, taken over all the circuit nodes.

$$\text{Effective Capacitance} = \sum_{\text{node } i} \eta_i \cdot c_i \quad (\eta_i: \text{activity of node } i, c_i: \text{physical capacitance of node } i)$$

The time duration values for each node are placed in the upper quadrant of the node. The latest arrival time, $arr(i)$, is placed in the left hand quadrant of node i and the earliest require time, $req(i)$, is placed in the right hand quadrant of node i . The time slack, $slack(i)$, for each node are placed in the lower quadrant of the node. The slack sensitive path is presented on its edges.

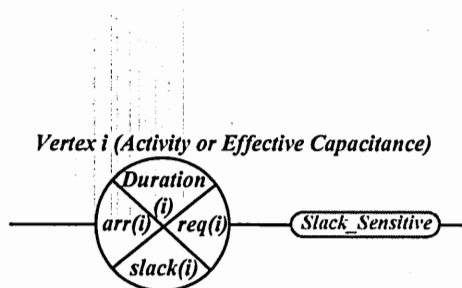


Figure 26. Graph-based timing symbol.

4.3 Energy and Delay Model

We use a transregional model for estimating the worst-case signal propagation delay through a gate. The delay model has been derived using an extension of the alpha-power law saturation drain current model [4-9] to the subthreshold region. The drain current model incorporates effects of high-field and quasi-ballistic (velocity overshoot) carrier transport in the MOSFET channel. The delay model consists of four major components: 1) the delay due to switching MOSFETs, 2) the distributed interconnect RC delay, 3) the time of flight delay, 4) the delay component due to the non-zero rise time of the input signal are considered. These definitions of gate delay and interconnect resistance delay allow the definition of arrival times and required times at the input and output of a gate in the network, which are used for defining time slack.

$$\begin{aligned}
 t_{d_v} = & \left[\frac{1}{2} - \frac{1 - \frac{V_{TS_i}}{V_{dd}}}{1 + \alpha} \right] \max_{i \in (1, f_{oi}(v))} \{t_{d_{i,v}}\} + \frac{V_{dd}/2}{I_{D_{vw}} - f_{ii}(v)\beta I_{off}} \cdot \left[C_{DP_v} + \frac{1}{w_v} \sum_{j=1}^{f_{oi}(v)} (w_{vj}C_{t_{vj}} + C_{INT_{vj}}) \right] \\
 & + \max_{j \in (1, f_{oi}(v))} \{t_{d_{v,j}}\} \left[R_{INT_{vj}} (w_{vj}C_{t_{vj}} + \frac{1}{2}C_{INT_{vj}}) + \frac{L_{INT_{vj}}}{v_{INT}} \right] + \frac{1}{2}C_{m_v}V_{dd} \sum_{j=1}^{f_{oi}(v)-1} \frac{1}{I_{D_{vw}}(j)} \quad (4-4)
 \end{aligned}$$

In the above equation, V_{dd} is the power supply voltage, t_{d_v} is the delay of gate G_v , V_{TS_i} is the threshold voltage of the i th gate, α is the velocity saturation coefficient ($1 \leq \alpha \leq 2$), $t_{d_{i,v}}$ is the delay of the gate G_v at the i th fan-in, $t_{d_{v,j}}$ is the delay of the gate G_v at the j th fan-out, $I_{D_{vw}}(f_{ii})$ is the switching drain current per unit width, f_{ii} is the number of

fanins, f_{oi} is the number of fanouts, β is the pMOS to nMOS width ratio ($\beta \geq 1$), I_{off} is the off current per unit width, C_{DP_v} is the sum of the overlap, junction and fringing capacitance at the output node per unit width, w_v is the device width, adjusting w_v scales the widths of all the transistors in G_v ($w_v \geq 1$), w_{vj} is the device width the gate at the j th fan-out ($w_{vj} \geq 1$), C_{tvj} is the input capacitance per unit width of the gate being driven by the j th fan-out, C_{INTvj} is the interconnect capacitance at the j th fan-out, R_{INTvj} is the interconnection resistance at the j th fan-out, L_{INTvj} is the interconnection length at the j th fan-out, v_{INT} is the propagation velocity through the interconnect, C_{mv} is the intermediate node capacitance of series connected MOSFET's in multiple fan-in gates, f_c is the clock frequency, η_v is activity factor of the gate output, and K_{SC} is the coefficient for short-circuit dissipation [4-10].

The equations used to compute the dynamic and static energy dissipations of a gate are described next. Similar models have been presented and analyzed in a recent work by [4-11]. It is assumed that the gates are simple multi-input gates with symmetric series or parallel pull-up and pull-down MOSFET configurations. Contributions of subthreshold leakage through the MOSFET channel as well as the leakage across the device drain junctions to static dissipation are included.

1) *Static Dissipation of Gate G_v ($v \in N$):*

$$E_{Static_v} = V_{dd} W_v I_{off} / f_c \quad (4-5)$$

2) *Dynamic and Short-Circuit Dissipation of G_v*

$$E_{Dynamic_v} = \frac{1}{2} \eta_v V_{dd}^2 (1 + K_{SC}) \cdot \left[w_v \{ C_{DP_v} + (f_{ii}(v) - 1) C_{mv} \} \sum_{j=1}^{f_{oi}(v)} (w_{vj} C_{tvj} + C_{INTvj}) \right] \quad (4-6)$$

4.4 Optimal Slack Distribution for Minimum Power

4.4.1 PREVIOUS APPROACHES

Before describing our algorithm, ZSA (Zero Slack Algorithm) [4-3] and MISA (Maximum Independence Set Algorithm) [4-6] are explained.

The ZSA algorithm starts from nodes with minimum slack and locally performs slack assignment such that the slack over the nodes is reduced to zero. This process iterated over all the nodes until the slack of all nodes is zero. More specifically, at each iteration, a node having the least positive slack (s_{\min}) is selected. A path in which all nodes are slack-sensitive to the selected node is then identified and the slack s_{\min} is distributed equally over all the nodes in that path. The procedure is repeated all such slack-sensitive paths. After all slacks have been updated, the procedure is repeated iteratively. The procedure of the ZSA algorithm is shown in Figure 27

Zero-Slack Algorithm (ZSA)

Input: directed acyclic graph $G = (V, E)$
Output: time slack distribution vector $TSD(v)$
Begin
 Phase 0: Initialization
 Initialize the class variables of all nodes V in G ;
 (delay, early start time, early finish time, late start time,
 late finish time, slack, level= ∞)
 Phase I: Making Timing Graph
 Phase II: Timing Analysis and Slack Calculation
 Sort all nodes according to the topological levels;
 For each node of the sorted V
 {
 Assign early start/finish times and late start/finish times;
 Compute time slacks;
 }
 Phase III: Distributing Time Slacks
 $S_{min} := \infty$;
 Find minimum positive slack, S_{min} ;
 $s := S_{min}$;
 For Slack-sensitive paths do begin
 {
 $\nabla s := s / (N_{min})$; N_{min} is the number of node on the path
 $delay(i) := delay(i) + \nabla s$;
 $s := s - \nabla s$;
 }
 Update $TSD(v)$
 Repeat **Phase II and III** until $S_{min} := \infty$;
End

Figure 27. Procedure of ZSA.

As opposed to ZSA, the MISA selects a node having the most positive slack (s_{max}). Then a maximum independence set (MIS) of a transitive slack-sensitive graph is extracted. MIS is a set of independent vertices (i.e., no two vertices are connected by an edge) such that the number of vertices in this set is maximum [4-6]. An incremental delay ($S_{max} - S_{(max-1)}$) is assigned to each node in MIS. This iteration is continued until all slack of the nodes are zero. However, *If the fan-out number is large enough*, the maximum total budget by MISA is as large as twice the maximum total budget by ZSA as shown in [4-6].

This shows that MISA provides significant improvement over ZSA when there exist a large number of fan-outs. The procedure of the MISA algorithm is shown in Figure 28.

Procedure *Maximum Independent Set Algorithm (MISA)*

Input: *directed acyclic graph $G = (V, E)$*
Output: *time slack distribution vector $TSD(v)$*
Begin

Phase 0: Initialization
Initialize the class variables of all nodes V in G ;
(delay, early start time, early finish time, late start time,
late finish time, slack, level= ∞)

Phase I: Making Timing Graph

Phase II: Timing Analysis and Slack Calculation
Sort all nodes according to the topological levels;
For *each node of the sorted V*
 {
 Assign early start/finish times and late start/finish times;
 Compute time slacks;
 }
Phase III: Distributing Time Slacks
Find maximum positive slack, S_{max} ;
While *($S_{max} > 0$)*
 {
 $\nabla s = \min\{\nabla s, S_{max} - S_{max-1}\}$;
 Find slack sensitive graph $G_{sub}(\nabla s)$ in G
 Find maximum independent set V_{mis} of $G_{sub}(\nabla s)$;
 Assign an incremental delay, ∇s , to each node in V_{mis} ;
 $delay(i) := delay(i) + \nabla s$, all i in V_{mis} ;
 Update $TSD(V)$ and S_{max} ;
 Repeat Phase II;
 }
End

Figure 28. Procedure of MISA.

However, the ZSA and the MISA are guaranteed that the speed performance improvement in placement, especially. Therefore, in this work, a power-aware version of ZSA (PA-ZSA) is proposed showing that the ZSA or the MISA is far from the optimal solution in terms of power.

4.5 Power Aware Zero Slack Algorithm (PA-ZSA)

In this section, we present an optimal slack-distribution algorithm for low power that generates a power-effective budget for any given directed acyclic graph G . Then, we compare the proposed algorithm with the conventional ZSA and MISA algorithms in terms of power savings

4.5.1 ALGORITHM DESCRIPTION

The proposed PA-ZSA algorithm begins with a set of vertices (circuit modules) with maximum *effective capacitance*. Next, a transitive slack-sensitive sub-graph of the circuit graph is constructed with the property that all the nodes of the sub-graph are slack-sensitive to the initial set of vertices that are chosen, based on the measure of effective capacitance. The slacks of the selected vertices are updated until all the slacks are reduced to zero. The primary problem is to guarantee that the slack distribution is optimal in terms of power and to reduce the complexity of exploring all circuit parameters for estimating the power and delay values corresponding to each vertex of the circuit DAG. A different metric for reassigning the slack values of all slack sensitive nodes at each iteration of the PA-ZSA algorithm is proposed. This metric minimizes the total power consumption over all the circuit nodes and is referred to as the energy-delay-ratio (EDR) metric. The original zero-slack algorithm is modified to accommodate this metric with additional objective of reducing all the slack values of the DAG nodes to zero.

In the PA-ZSA algorithm, initially, *Monte Carlo simulation* is performed to determine the activity profile of each module/sub-module. This consists of applying randomly generated input patterns at the primary inputs of the circuit and monitoring the switching activity per time interval T using a switching activity simulator. Under the assumption that the switching activity of a circuit module over any period T has a normal distribution, and for a desired percentage error in the activity estimate and a given confidence level, the number of required simulation vectors is estimated. The simulation based approach is accurate and capable of handling various device models, different circuit design styles, single and multi-phase clocking methodologies, etc. While other more advanced techniques [4-1,4-2] may be employed to determine the switching activities, this is not the primary focus of this paper and is not discussed here further.

In the phase I of the PA-ZSA algorithm (see Figure 29), the circuit topology is mapped to a DAG by using a breath-first-search based algorithm. Then, slack times are calculated via Equations (1~3) in phase II. Finally, in phase III, the surplus time slacks are distributed iteratively according to the *Assign_EDR* metric function (see Figure 30) until all slacks of the nodes are zero.

Definition 6: Let the slack associated with module i be given by ∇s_i . For N modules, the *slack gradient vector*, ∇s , is defined to be $\nabla s = [\nabla s_1, \nabla s_2, \nabla s_3, \dots, \nabla s_{N-1}, \nabla s_N]$.

Definition 7: The $G_{sub}(v,e)$ is the slack sensitive sub-graph where the vector v is the nodes and the vector e is the edges in G_{sub} after timing analysis and slack calculation.

Definition 8: Let the v be the subset vector in $Gsub(v, e)$ after timing analysis and slack calculation. For N modules in v , $TSD(v)$ is defined to be by the time slack distribution of the subset vector v .

Definition 9: $Amax(i,j)$ is defined by a maximum effective capacitance pair where the node i is the node in $Gsub(v,e)$ with highest effective capacitance and the node j is the node in $Gsub(v,e)$ with the second highest effective capacitance.

Power-Aware Zero-Slack Algorithm (PA-ZSA)

```

Input: directed acyclic graph  $G = (V,E)$ 
Output: Power-aware surplus (increased) slack gradient vector  $\nabla s$ ;
Begin
  /* Phase 0: Initialization */
  Initialize variables of all nodes  $V$  in  $G$ ;
  (delay, activity, early start time, early finish time, late start time,
  late finish time, slack, topology level)
  Initialize time slack distribution vector  $TSD(v)$ ;
  Initialize surplus (increased) slack gradient vector  $\nabla s$ ;
  /* Phase I: Making Timing Graph */
  While (  $TSD(v)$  is not zero )
  {
    /* Phase II: Timing Analysis and Slack Calculation */
    Sort all nodes according to the topological levels;
    For each node of the sorted  $v$ 
    {
      Assign early start/finish times and late start/finish times;
      Compute time slacks;
    }
    /* Phase III: Distributing Time Slacks */
    Find slack sensitive graph  $Gsub$  in  $G$ ;
    While ( $Gsub$  is not empty )
    {
       $\nabla s = Assign\_EDR(\nabla s)$ ;
       $delay(i) := delay(i) + \nabla s$ , all  $i$  in  $Gsub$ ;
      Update  $TSD(v)$  and  $Gsub$ ;
    }
  }
End

```

Figure 29. Proposed PA-ZSA algorithm.

After obtaining the slack sensitive sub-graph, $Gsub(v,e)$, and surplus slack gradient vector, ∇s , the surplus slack is distributed according to the EDR metric as shown

in Figure 30. First the maximum effective capacitance set A_{max} (node i , node j) is extracted. If the node i and the node j is slack-sensitive each other, the delay i and the delay j are respectively assigned according to the cost function of $\frac{Effective_Capacitance(E_i)}{delay_i} = \frac{Effective_Capacitance(E_j)}{delay_j}$. In other words, the delay of the node i becomes $delay_i = delay_i + (slack_i + slack_j) \cdot \frac{E_i}{E_i + E_j}$ within the slack gradient (∇s_i) of the node i , and the delay of node j becomes $delay_j = delay_j + (slack_i + slack_j) \cdot \frac{E_j}{E_i + E_j}$ within the slack gradient (∇s_j) of the node j . If a node is not slack-sensitive to another node, the delay of the node is increased up to the slack gradient (∇s) of the node.

Assign_EDR (∇s)

```

Input: Gsub = (V,E)
Output: power-aware surplus slack gradient vector  $\nabla s$ 
Begin
  Find maximum effective capacitance pair-wise set  $A_{max}$  in Gsub;
   $A_{max} = \{node\ i, node\ j\}$ , where
    node  $i$  = node in Gsub with highest effective capacitance,
    node  $j$  = node in Gsub with second highest effective cap;
  If node  $i$  and  $j$  is slack-sensitive each other
  {
    delay  $i$  = min [ delay  $i$  + (slack  $i$  + slack  $j$ ) *  $\frac{effective\ cap\ i}{(effective\ cap\ i + effective\ cap\ j)}$  ,  $\nabla s_i$  ]
    delay  $j$  = min [ delay  $j$  + (slack  $i$  + slack  $j$ ) *  $\frac{effective\ cap\ j}{(effective\ cap\ i + effective\ cap\ j)}$  ,  $\nabla s_j$  ]
  }
  else
  {
    delay  $i$  = delay  $i$  + slack  $i$ ;
    delay  $j$  = delay  $j$  + slack  $j$ ;
  }
  Update  $\nabla s$ ;
End

```

Figure 30. Assign_EDR algorithm.

The EDR-based slack distribution is explained in *Lemma 4.1*, below. It is assumed that the energy consumptions of the different modules in the DAG can be

adjusted by modulating the supply voltage to each module. Subsequently, it is shown *empirically* that the EDR metric yields near-optimal power consumption for the case in which other technology features such as threshold voltage are also used to modulate the power and delay of each module. ¹

Lemma 4.1: EDR (Energy-Delay Ratio) Paradigm: When the ratios of the energy and the delay for each module are same, the total energy consumption is minimal. In other words, the delay for each module should be proportional to the energy of that module for minimum power consumption. Therefore the surplus time slacks should be assigned onto each module according to the cost function of $\frac{E_1}{d_1} = \frac{E_2}{d_2} = \dots = \frac{E_n}{d_n}$ (1,..., n: module number) for low power.

Proof: Let us assume the energy consumptions of the module1 (M1) and module2 (M2) in Figure 31 are $E_1 = K_1 V_{dd1}^2$ and $E_2 = K_2 V_{dd2}^2$, respectively. Here, the K_1 and K_2 are proportional to the switching activities and load capacitances of the modules. The delay of the M1 can be assumed by $d_1 = L_1 / V_{dd1}$ and the delay of the M2 can be assumed by $d_2 = L_2 / V_{dd2}$, where the L_1 and L_2 are dependant on the threshold voltage of each module. And the total delay is $T = d_1 + d_2$. We want to minimize the total power P_{Total} :

$$P_{Total} = \frac{K_1 V_{dd1}^2 + K_2 V_{dd2}^2}{T} \quad (4-7)$$

¹ For simplicity, we do not consider power-delay models for the logic level shifters that are necessary with the use of multiple supply voltages. These can, however, be incorporated easily into the proposed optimization algorithms.

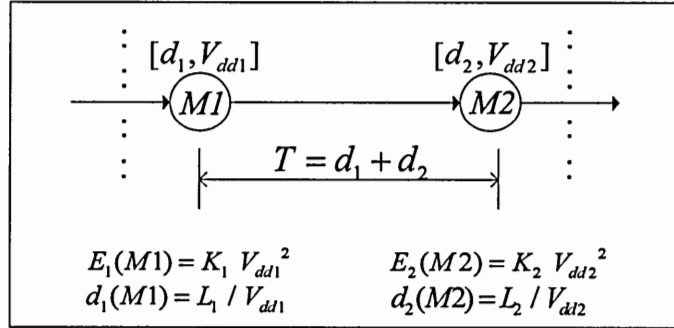


Figure 31. Two nodes example.

Now,

$$d_2 = T - d_1 = \left(T - \frac{L_1}{V_{dd1}}\right) = \left(\frac{TV_{dd1} - L_1}{V_{dd1}}\right) \quad (4-8)$$

and after transposing, we have $V_{dd1} = \left(\frac{L_1}{T - d_2}\right)$ and $V_{dd2} = \left(\frac{L_2}{d_2}\right)$. Let us substitute for

V_{dd1} and V_{dd2} , then the total power P_{Total} is

$$P_{Total} = \frac{K_1 \left(\frac{L_1}{T - d_2}\right)^2 + K_2 \left(\frac{L_2}{d_2}\right)^2}{T} \quad (4-9)$$

$$\frac{\partial P_{Total}}{\partial d_2} = \left(\frac{K_1 L_1^2}{T} \times (-2) \times \frac{1}{(T - d_2)^3} \times (-1)\right) - \left(\frac{2K_2}{T} \cdot \frac{L_2^2}{d_2^3}\right) \quad (4-10)$$

At minimum power, $\frac{\partial P_{Total}}{\partial d_2} = 0$, therefore,

$$\left(\frac{K_1 L_1^2}{T} \times (-2) \times \frac{1}{(T-d_2)^3} \times (-1) \right) - \left(\frac{2K_2 \cdot L_2^2}{T \cdot d_2^3} \right) = 0 \quad (4-11)$$

$$\frac{K_1 L_1^2}{K_2 L_2^2} = \frac{(T-d_2)^3}{d_2^3} \text{ or } \frac{(T-d_2)}{d_2} = \left(\frac{K_1 L_1^2}{K_2 L_2^2} \right)^{1/3} \quad (4-12)$$

Let

$$\alpha = \left(\frac{K_1 L_1^2}{K_2 L_2^2} \right)^{1/3} \quad (4-13)$$

, then $\frac{T}{d_2} = \alpha + 1$. So, when P_{Total} is minimum,

$$d_2 = \frac{T}{\alpha + 1} \quad (4-14)$$

$$d_1 = T - d_2 = \frac{\alpha T}{\alpha + 1} \quad (4-15)$$

Recall $E_1 = K_1 V_{dd1}^2$ and $E_2 = K_2 V_{dd2}^2$, then,

$\frac{E_1}{d_1} = \frac{K_1 V_{dd1}^2}{d_1} = \frac{K_1 L_1^2}{d_1^3}$. Substituting for d_1^3 with optimal d_1 from Equation (4-15),

$$\frac{E_1}{d_1} = \frac{K_1 L_1^2}{\alpha^3 T^3} (\alpha + 1)^3 \quad (4-16)$$

$\frac{E_2}{d_2} = \frac{K_2 V_{dd2}^2}{d_2} = \frac{K_2 L_2^2}{d_2^3}$ and Substituting for d_2^3 with optimal d_2 from Equation (4-14),

$\frac{E_2}{d_2} = \frac{K_2 L_2^2}{T^3} (\alpha + 1)^3$. But from Equation (4-13), we know that $K_2 L_2^2 = \frac{K_1 L_1^2}{\alpha^3}$, so

$$\frac{E_2}{d_2} = \frac{K_1 L_1^2}{\alpha^3 T^3} (\alpha + 1)^3 \quad (4-17)$$

The Equation (4-16) and Equation (4-17) are exactly same. Hence, for total minimum power,

$$\frac{E_2}{d_2} = \frac{E_1}{d_1}$$

□

In the following validation, an experimental proof of EDR is illustrated when simultaneous supply voltage, threshold voltage, and device width optimization is applied to an accuracy-proven complex device model [4-9].

Empirical validation of EDR metric for more complex technology mappings: An experiment was conducted to see if the EDR paradigm holds up when supply voltage, threshold voltage and device sizing are used to control the power consumption of the two modules in Figure 32. A circuit power optimizer was used to optimize the designs of modules 1 and 2 of the figure, which were chosen to be a 8X4 MUX and a 4-BIT ADDER, respectively. The tool takes as input, the circuit description and allowed maximum delay of each module, *optimizes the power supply voltage, threshold voltage, and device sizes of the gates of that module* for minimum power. The optimizer uses the alpha-power law MOSFET model for power-delay-sizing optimization. A range of delay

values were assigned to the MUX and the ADDER in such a way that the total delay is up to 8ns. EDR-based metric is shown in Equation (4-18). The graph in Figure 33 shows that the total power consumption of the circuit is minimal when the value of the metric is close to 1. This shows that the EDR paradigm holds up when very complex power optimizations are applied to complex device models.

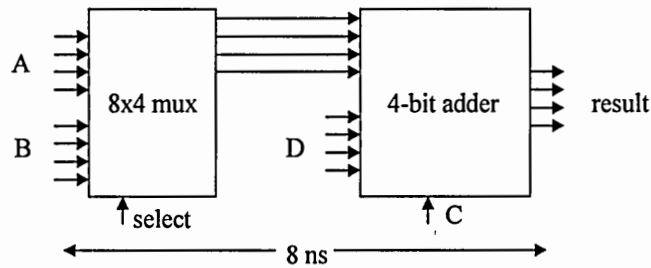
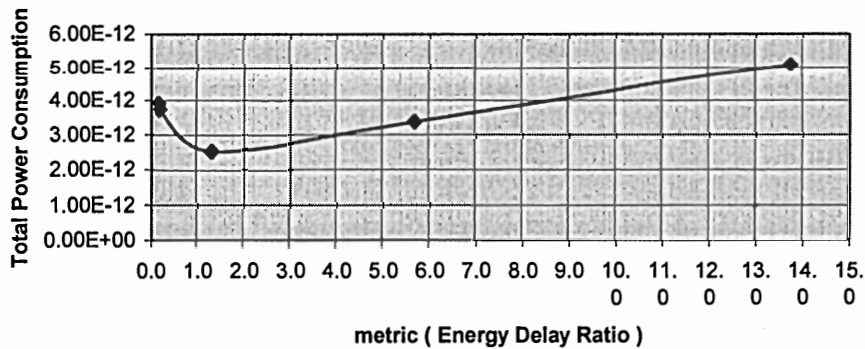


Figure 32. Two modules example.

$$Metric = \frac{Delay_{adder} / Delay_{mux}}{Energy_{adder} / Energy_{mux}} \quad (4-18)$$



d(adder)	P(adder)	d(mux)	P(mux)	d(total)	P(total)	metric
5.00E-09	3.58E-12	2.36E-09	3.35E-13	7.37E-09	3.91E-12	0.20
5.31E-09	3.40E-12	2.36E-09	3.35E-13	7.68E-09	3.73E-12	0.22
6.39E-09	1.92E-12	1.59E-09	6.34E-13	7.98E-09	2.55E-12	1.33
6.75E-09	1.85E-12	9.71E-10	1.52E-12	7.72E-09	3.37E-12	5.71
6.94E-09	1.84E-12	8.96E-10	3.26E-12	7.84E-09	5.10E-12	13.77

Figure 33. EDR validation.

4.5.2 COMPARISON WITH ZSA AND MISA

The power optimization capability of PA-ZSA vs. ZSA and MISA is demonstrated by application to the benchmark circuit c17 as shown in Figure 34. Figure 35 presents the DAG graph mapping and Figure 36 shows the results of initial timing and slack analysis for the circuit of Figure 34. Note that additional vertices 1, 2, 3, 6 and 9 are inserted into the DAG to represent the circuit inputs and the activities on those input lines. Figure 37 shows the slack-sensitive subgraph of the DAG of Figure 36 obtained by removing all vertices with zero initial slack.

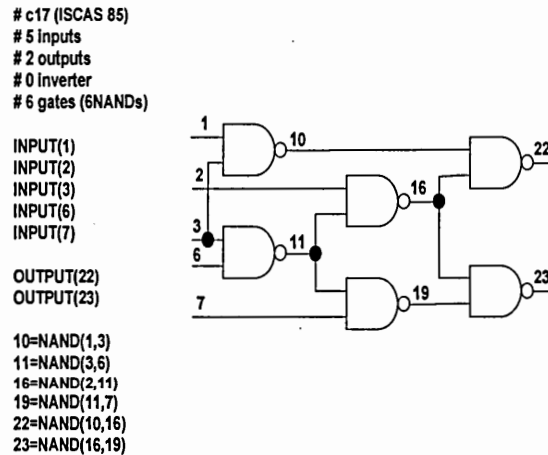


Figure 34. An example circuit (c17).

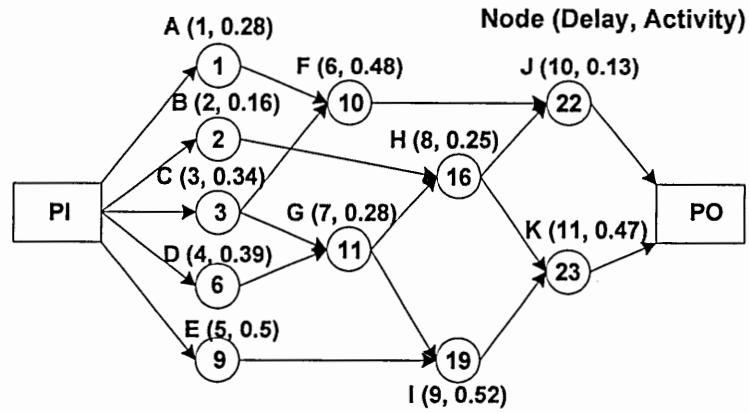


Figure 35. DAG mapping.

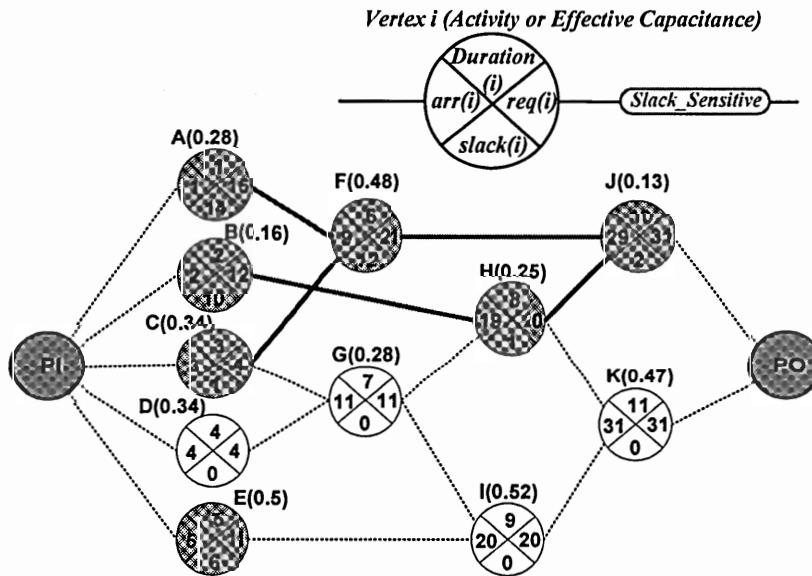


Figure 36. Timing and slack analysis.

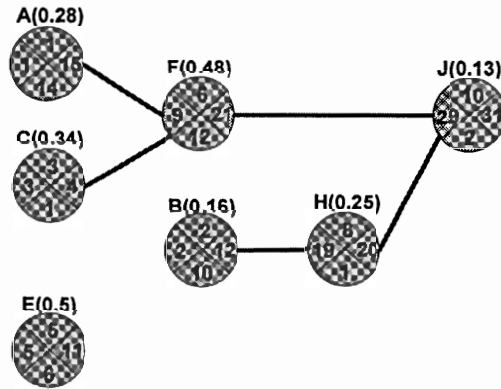


Figure 37. Slack-sensitive paths.

According to the EDR metric, modules with higher effective capacitance should be assigned correspondingly higher delay for minimal overall power consumption. This indicates that the metric *power budget* as shown in below is a good indicator of the efficacy of one slack assignment algorithm over another.

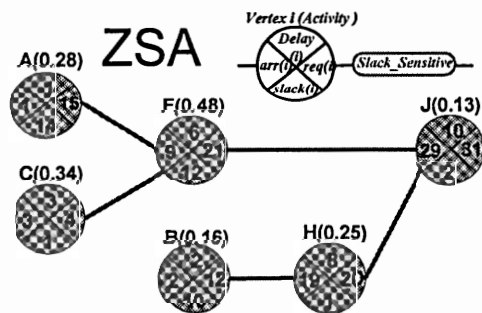
$$\text{Power Budget} = \sum_{\text{node } i} \eta_i \cdot c_i \cdot \nabla s_i \quad (\eta_i: \text{activity of node } i, c_i: \text{physical cap of node } i, \nabla s_i: \text{surplus slack of node } i)$$

In other words, a slack assignment algorithm that maximizes this metric over another such algorithm is better from the viewpoint of power minimization for the given circuit. Hence, this metric is used to compare PA-ZSA vs. ZSA and MISA. In the following, we first explain the operation of ZSA and MISA on the circuit of Figure 34. Then we show how the proposed PA-ZSA approach operates on the same circuit.

Operation of ZSA:

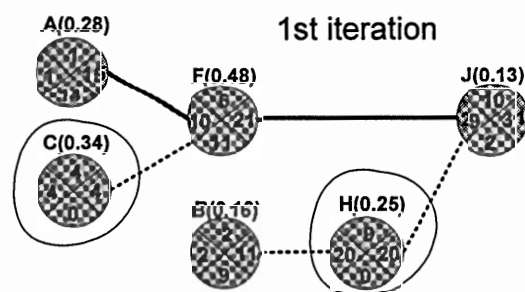
ZSA starts with nodes with minimum slack. In the example of Figure 38(a), nodes C and node H have minimum positive slack (s_{min}). In the first iteration, the

incremental delays ($s_{min}/N_{min}=1/1=1$) for node C and H are assigned respectively, where N_{min} is the number of nodes in the slack-sensitive path. After slack distribution and recalculation in the first iteration, the graph of Figure 38(b) is obtained. The delay of the node C is increased from 3 to 4 and its slack becomes zero and similarly the delay of the node H is increased from 8 to 9 and its slack becomes zero. In the next iteration, the vertex with the minimum positive slack is identified again and slack re-assignment is performed iteratively until all the nodes' slacks become zero. Figure 38(c-e) shows the iterations. In the final iteration, shown in Figure 38(e), we can see that all slacks on the nodes are zero and the slack gradient vector is $\nabla S = \{3,1,9,9,1,2\}$ corresponding to the node vector $V=\{A,C,F,B,H,J\}$ (the slack gradient vector is the vector of changes in the slacks of modules A, C, F, B, H and J, respectively due to application of the ZSA algorithm). The total slack-budget is 25 and the **power budget** is 7.45 as shown in Figure 38 (f).



vertex	A	C	F	B	H	J
slack	14	1	12	10	1	2

(a) Original timing graph



vertex	A	C	F	B	H	J
slack	14	0	11	9	0	2

(b) 1st iteration

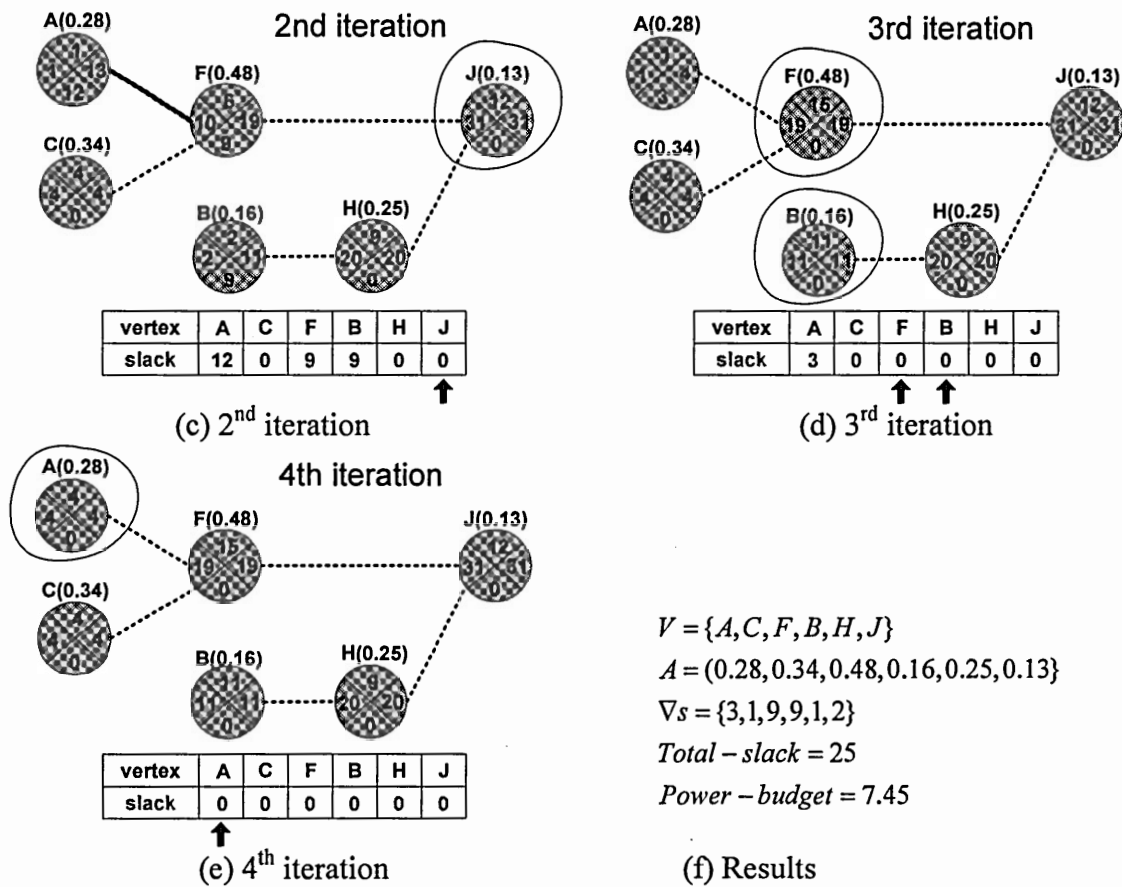
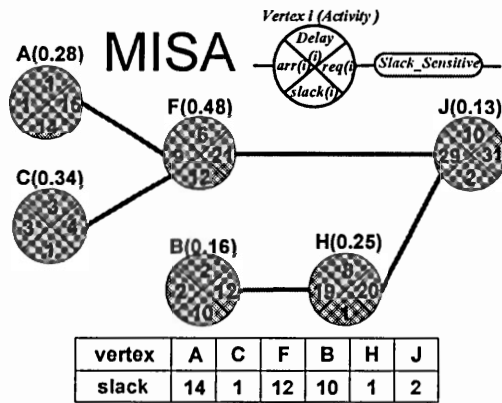


Figure 38. Slack distribution of ZSA.

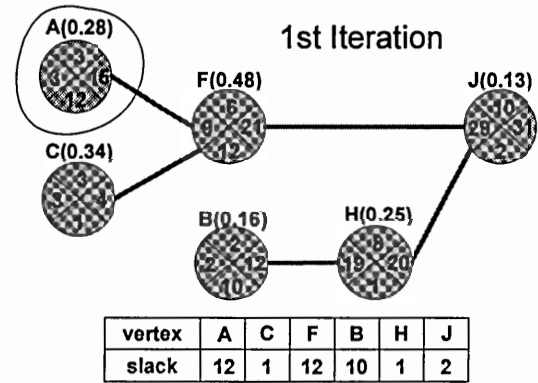
Operation of MISA :

As opposed to ZSA, MISA selects a node having the most positive slack (s_{max}). Therefore, first, node A is selected for slack re-distribution as it has the largest initial slack ($s_{max}=14$). An incremental delay ($S_{max}-S_{(max-1)}= 14-12 =2$) is assigned to node A by the MISA algorithm as shown in Figure 39(b). This procedure (iterations 2-6) is continued until all the slacks of the nodes are zero as shown in Figure 39(c-g). As seen in Figure 39(h), we can see that all the slacks of the nodes become zero and the slack gradient vector due to application of the MISA algorithm is $\nabla S = \{11,1,1,10,0,2\}$

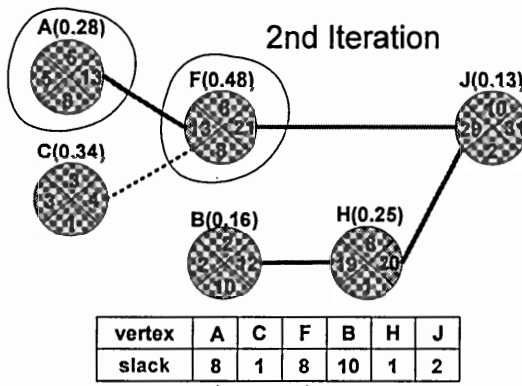
corresponding to the node vector $V=\{A,C,F,B,H,J \text{ as before}\}$. The total-slack budget is 25 and the *power budget* is 6.24 as shown in Figure 39(h).



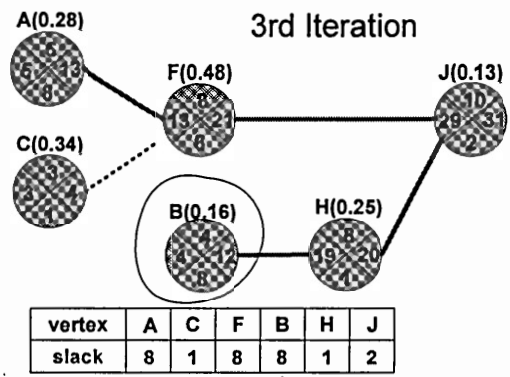
(a) Original timing graph



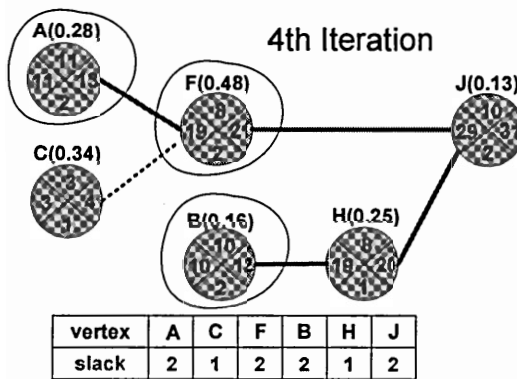
(b) 1st iteration



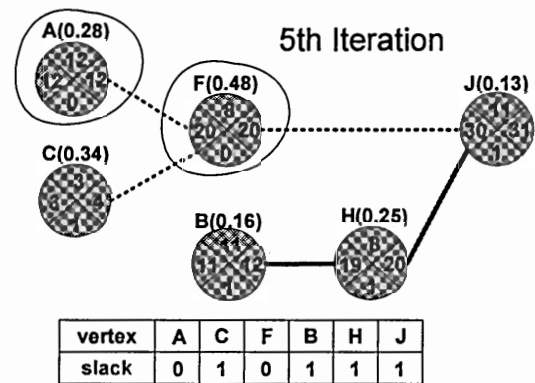
(c) 2nd iteration



(d) 3rd iteration



(e) 4th iteration



(f) 5th iteration

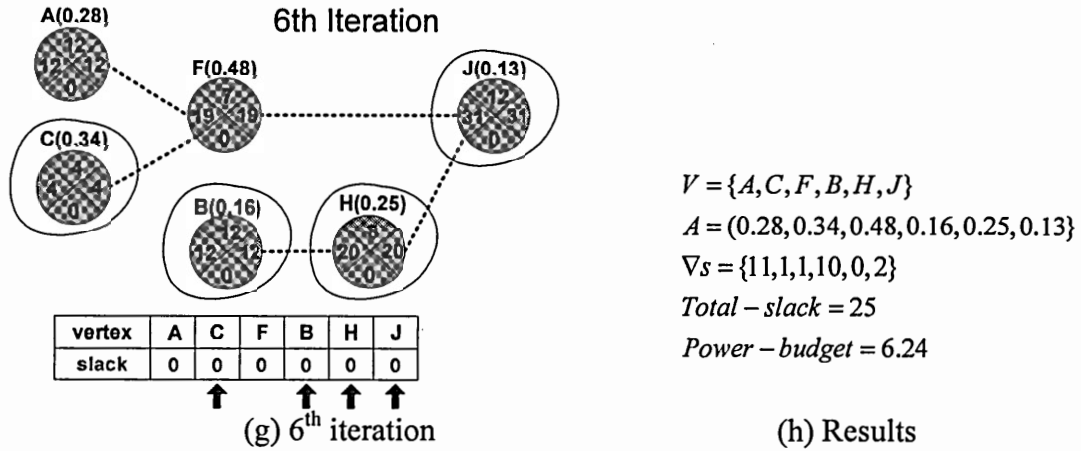


Figure 39. Slack distribution of MISA.

Operation of PA-ZSA:

PA-ZSA begins with a set of nodes having the most activity (A_{max}). In this example, nodes F and node C have the most activities ($Act(F)=0.48$ and $Act(C)=0.34$) and these two nodes are also slack-sensitive. Therefore, the transitive sub-graph $G_{sub} = (A_{max}, E_{sub})$ is constructed, where $A_{max} = \{F, C\}$. An incremental delay ∇s for each node in A_{max} is assigned according to the EDR metric and the maximum slack budget of the nodes in A_{max} are as shown in Figure 40(b). This procedure is continued iteratively until all the slacks of all the nodes are zero as shown in Figure 40(c-d). As shown in Figure 40(d), we can see that all slacks on the nodes become zero and the slack gradient vector due to application of PA-ZSA is $\nabla s = \{3, 1, 10, 9, 1, 1\}$ corresponding to the node vector $V = \{A, C, F, B, H, J\}$. The total-slack budget is 25 and the *power budget* is 7.8 as shown in Figure 40(e).

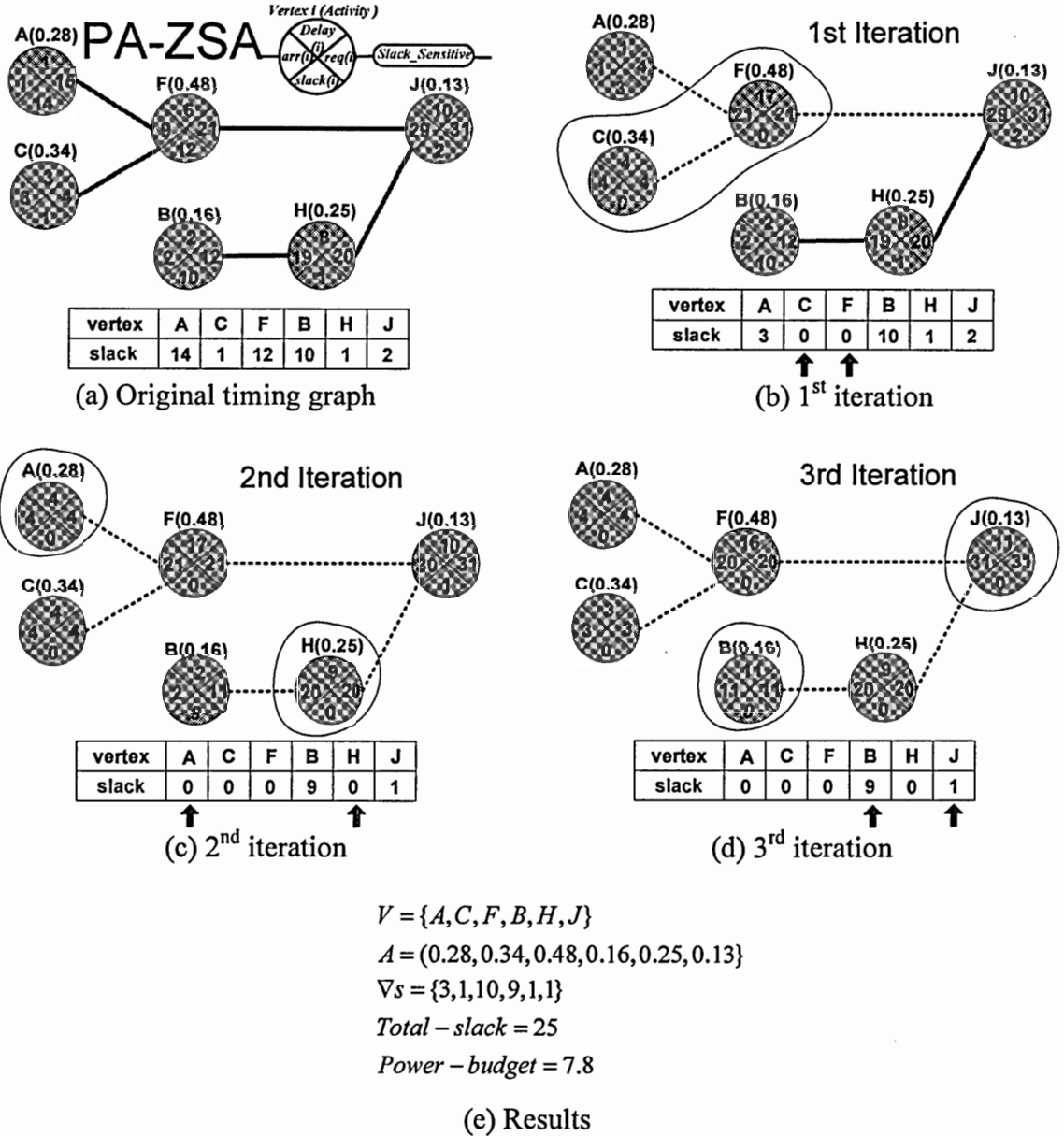


Figure 40. Slack distribution of PA-ZSA.

In this section, we compared PA-ZSA with the ZSA and MISA algorithms in terms of the power-budget (power-saving possibility). Table 9 summarizes the results for different slack distribution algorithms for the previous example. The proposed PA-ZSA has more power-saving possibility than other algorithms.

Table 9. Power budget comparison.

Algorithm	ZSA	MISA	PA-ZSA
Power-Budget	$\nabla s = \{3, 1, 9, 9, 1, 2\}$ <i>Total - slack = 25</i> <i>Power - budget = 7.45</i>	$\nabla s = \{1, 1, 1, 1, 10, 0, 2\}$ <i>Total - slack = 25</i> <i>Power - budget = 6.24</i>	$\nabla s = \{3, 1, 10, 9, 1, 1\}$ <i>Total - slack = 25</i> <i>Power - budget = 7.8</i>

4.5.3 TIME COMPLEXITY COMPARISON

At each iteration, the most time consuming operation is computation of slacks for the circuits. Complexity of the PA-ZSA algorithm is $O(nb^m)$, where n is the number of vertices, b is the branching factor (i.e., average fan-out number of the interconnections) and m is maximum topological level. Table 10 shows time-complexity comparison with ZSA and MISA.

Table 10. Complexity comparison.

Algorithm	ZSA	MISA	PA-ZSA
Complexity	$O(ke)$	$O(n \log(n^2/e))$	$O(nb^m)$

Where, k is iteration number, e is number of edges, n is number of vertices, b is branching factor, and m is topological depth.

4.6 Gate-Level Power Optimization

After the maximum delays have been assigned to each module/gate in the circuit, we optimize each gate individually for minimum power. The strategy is to find iteratively, using binary search, the optimal combination of Vdd, Vth, and W for each gate that meets

the maximum delay condition while achieving minimum power dissipation. We used our previous work for the gate level power optimization [4-7]. This strategy is based on the observation that power consumption and delay are monotonic functions of Vdd, Vth, and W, individually, other parameters being fixed. Since it is impractical to have more than one power supply or threshold voltage in the circuit, we keep only one global value of Vdd and Vth. However, the algorithm could be easily modified to allow the use of multiple threshold values in the circuit if desired. The algorithmic complexity of this procedure depends on the number of iteration steps that we allow for convergence to the optimal values. Assuming that Vdd, Vth and W are each constrained to 2^M quantized values, it takes $O(M^3)$ simulations of the entire circuit to obtain the final optimal values. This is many orders of magnitude lower than the complexity of any direct or random search algorithm that may be used to search for the optimal solution. The procedure for this optimizer is shown in Figure 41.

Procedure Gate-Level Power Optimizer

```
VddRange ← [0.6v-1.2v]
For M steps do
  Vdd ← MID(VddRange)
  VthRange ← [0.1v-0.7v]
  For M steps do
    Vth ← MID(VthRange)
    For all gates do
      WRange ← [1-100]
      For M steps do
        If delay(g) smaller than max delay(g), then
          WRange ← LOWER(WRange)
        else
          WRange ← HIGHER(WRange)
        endif
      endFor
    endFor
    If All g: delay bounded and total energy decreased
    then
      VthRange ← HIGHER(VthRange)
    else
      VthRange ← LOWER(VthRange)
    endif
  endFor
  If All g: delay bounded and total energy decreased
  then
    VddRange ← LOWER(VddRange)
  else
    VddRange ← HIGHER(VddRange)
  endif
endFor
```

Figure 41. Gate-level power optimizer.

4.7 Experimental Results

We used several tools for the RTL description, the functional verification and the logic synthesis and developed some interface programs and simulators for the proposed hierarchical optimization with C/C++/STL on Ultra-80 Unix machine. Some arithmetic modules are used for the benchmark circuits. For the range of the technology parameter values, we refer to the 2001 updated version of ITRS (International Technology Roadmap for Semiconductors). We used verilog for the RTL design, VCS (synopsys) for

the verilog functional simulation, and design analyzer (synopsys) with 0.25 micron TSMC library for the logic synthesis. Figure 42 shows an overall procedure of the optimization methodology.

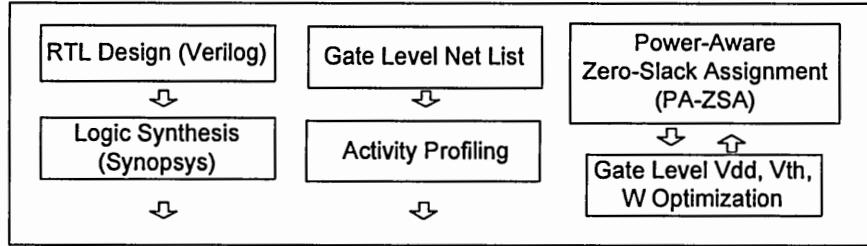


Figure 42. Optimization procedure.

Table 11 demonstrates the impact of the PA-ZSA based optimization in terms of power and area. We synthesized few arithmetic functional units from ARM core and then, assigned slack budgets and optimized through our gate-level optimization tool. Table 11(a) shows power dissipation when we used fixed threshold voltage, optimal supply voltage, and optimal device width as a reference before optimization. First column of the table shows the system modules used in this simulation, second column presents the number of gates and topological level (depth) of the circuits, third column shows the delay constraints, fourth column shows different input activities (high and low activity cases), fourth and fifth column shows optimal supply voltage and device width, and finally static (leakage), switching, short-circuit, and total power is presented in the next columns. Table 11(b) shows power dissipations after optimization with optimal values of supply voltage, threshold voltage, and device width operating with the same circuit speed. We can obtain average 2x – 10x power savings of total (dynamic and static) power. Table 12 shows the effectiveness of the proposed slack budget distribution scheme over

traditional algorithms with same gate-level optimization and different slack budget mechanisms. Table 12(a) shows the power dissipation results by using ZSA algorithm with a 64-bit ALU circuit, Table 12(b) shows the power dissipation results by using MISA algorithm, and Table 12(c) shows the power consumption results by using proposed PA-ZSA algorithm. Approximately 17%-29% more power reduction over conventional slack budget distribution algorithms is achieved.

Table 11. Results of PA-ZSA based optimization.

(a) Before Optimization [Fixed Vth (0.7v), Optimal Vdd (0.6-1.2v), Optimal W (1-100u)]

System Module	Gates/Depth	Delay (ns)	Input Activity	Vdd	Power Dissipation				
					W min, avg, max	Static	Dynamic	Short-ckt	Total
4 - Full Adder	106/48	3.36	0.5	0.725	2, 15, 24	2.19x10E-15	1.45x10E-11	2.22x10E-12	1.67x10E-11
			0.05	0.75	2, 15, 20	2.19x10E-15	1.33x10E-13	2.95x10E-14	1.65x10E-13
16- Full Adder	1030/93	6.98	0.5	0.725	2, 45, 80	9.78x10E-15	7.45x10E-11	4.56x10E-12	7.91x10E-11
			0.05	0.725	2, 42, 68	9.78x10E-15	7.98x10E-13	8.74x10E-13	8.74x10E-13
16 - Look ahead (1)	1838/81	7.0	0.5	0.725	12, 19, 56	8.02x10E-15	4.95x10E-11	2.26x10E-12	5.17x10E-11
			0.05	0.725	10, 18, 54	8.02x10E-15	1.98x10E-13	3.86x10E-14	2.45x10E-13
16 - Look ahead (2)	1871/75	6.3	0.5	0.75	2, 24, 48	7.82x10E-15	3.45x10E-11	2.09x10E-12	3.66x10E-11
			0.05	0.75	2, 24, 64	7.82x10E-15	1.91x10E-13	3.00x10E-14	2.29x10E-13
16 - Look ahead (3)	1928/69	5.9	0.5	0.75	8, 16, 45	5.88x10E-15	2.75x10E-11	1.15x10E-12	2.86x10E-11
			0.05	0.75	8, 14, 32	5.88x10E-15	1.80x10E-13	1.66x10E-14	2.02x10E-13

(b) After Optimization [Optimal Vth (0.1-0.7v), Optimal Vdd (0.6-1.2v), Optimal W (1-100u)]

System Module	Gates/Depth	Delay (ns)	Input Activity	Vdd, Vth	W	Total Power	Power Saving (x)
					min, avg, max		
4 - Full Adder	106/48	3.36	0.5	0.625, 0.2	2, 6, 12	6.48x10E-12	2.57x
			0.05	0.625, 0.2	2, 8, 14	7.57x10E-14	2.17x
16- Full Adder	1030/93	6.98	0.5	0.65, 0.2	2, 20, 42	9.16x10E-12	8.63x
			0.05	0.65, 0.2	2, 16, 24	1.96x10E-13	4.45x
16 - Look ahead (1)	1838/81	7.0	0.5	0.6, 0.1	2, 6, 14	5.15x10E-12	10.04x
			0.05	0.6, 0.1	2, 6, 14	1.40x10E-13	1.75x
16 - Look ahead (2)	1871/75	6.3	0.5	0.65, 0.15	2, 10, 14	4.20x10E-12	8.71x
			0.05	0.65, 0.125	2, 8, 14	7.04x10E-14	3.25x
16 - Look ahead (3)	1928/69	5.9	0.5	0.65, 0.12	2, 7, 18	3.18x10E-12	9.00x
			0.05	0.65, 0.12	2, 6, 13	7.06x10E-14	2.86x

Table 12. Comparison with other ZSAs.

(a) ZSA Based Optimization (Vdd:0.6-1.2v, Vth:0.1-0.7v)						
System Module	Delay (ns)	Gates/ Depth	Input Activity	Vdd, Vth	W	Total Power
					min, avg, max	
64 - ALU	20.07	3417/ 226	0.5	0.625, 0.1	2, 6, 22	2.93x10E-9
			0.05	0.625, 0.1	2, 8, 24	5.82x10E-10
(b) MISA Based Optimization (Vdd:0.6-1.2v, Vth:0.1-0.7v)						
System Module	Delay (ns)	Gates/ Depth	Input Activity	Vdd, Vth	W	Total Power
					min, avg, max	
64 - ALU	20.07	3417/ 226	0.5	0.625, 0.1	2, 15, 24	3.88x10E-09
			0.05	0.625, 0.1	2, 15, 20	6.81x10E-10
(c) PA-ZSA Based Optimization (Vdd:0.6-1.2v, Vth:0.1-0.7v)						
System Module	Delay (ns)	Gates/ Depth	Input Activity	Vdd, Vth	W	Total Power
					min, avg, max	
64 - ALU	20.07	3417/ 226	0.5	0.625, 0.1	2, 4, 14	2.07x10E-09
			0.05	0.625, 0.1	2, 5, 12	4.82x10E-10

*Note: Scheme (c) is better than scheme (a) around 17.9% - 29.5% in total power reduction

4.8 Summary

In this chapter, we proposed a slack distribution algorithm for ultra low-power CMOS logic circuits in a VLSI design environment. We introduced *Power-Aware Zero-Slack Algorithm* (PA-ZSA), which distributes the surplus time slacks into the most power-hungry modules. Based on these time slacks, we have conducted the low-power optimization at gate level by using technology scaling technique. We have tested the proposed algorithms on a set of benchmark circuits and some building blocks of synthesizable ARM core. The experimental results show that our polynomial-time solvable strategy delivers an order of magnitude savings in total (static and dynamic) power over the design without power optimization and our slack distribution strategy reduces more power over the conventional slack budget algorithms with the same power optimization.

CHAPTER 5

HIERARCHICAL ACTIVITY-AWARE DELAY ASSIGNMENT FOR LOW POWER

5.1 Introduction

Recent advances in wireless networking technology and the rapid development of semiconductor technology have introduced new challenges in the design of portable devices such as personal digital assistants (PDAs). Power optimization for those embedded systems and power constrained mobile computing is an active area of research that has received considerable attention in most recent years. Delay, area and power trade-offs for complex systems require the use of advanced algorithms and EDA tools. To achieve excellent power and performance results, future EDA tools must harness the combination of technology parameters, i.e., multiple supply voltages (V_{dd}), multiple threshold voltages (V_{th}), and transistor resizing (W). By combining the optimization strategy with the on-the-fly technology parameter scaling, designers and EDA tools can fully explore the design space of dynamic power, static power, and timing slack [5-1,5-2].

In this chapter, we introduce the *Hierarchical Activity-Aware Time Slack Distribution* (HA²TSD) algorithm, which distributes the surplus time slack into the most power-hungry modules hierarchically as shown in Figure 43. HA²TSD ensures that the total slack budget is maximal and the total power is near-minimal. Based on these time slacks, we have optimized technology parameters (supply voltage, threshold voltage, and device width) through a gate-level power optimizer and have tested the algorithm on a set of benchmark example circuits and building blocks of a synthesizable ARM core.

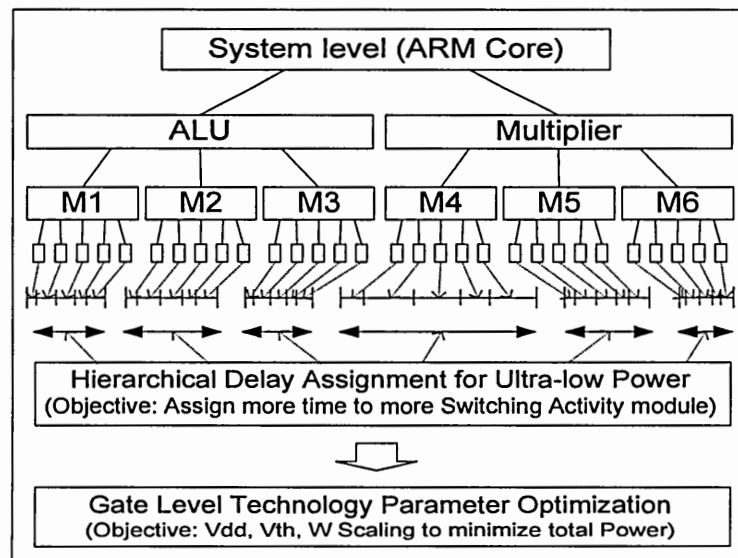


Figure 43. Hierarchical delay assignment and power optimization.

5.2 Hierarchical Activity-Aware Delay Assignment

The key steps of our approach are shown in Figure 44. First hierarchical circuit partitioning is performed. Then, beginning with the topmost level of the design hierarchy,

delay values are assigned to every module at that level. The total delay from PI to PO is given. The problem is to determine the delays of the individual modules so that total power consumption can be minimized by optimizing the supply voltage, threshold voltage and device sizes of module M_j for the assigned delay values. The procedure is repeated hierarchically. We use the following heuristic to assign delays to each module.

In a given data flow graph of M_j modules, let $C_j = \sum_{\text{node } i} \eta_i c_i$ be the summation of the product of the activity η_i at node i and the capacitance c_i at node i over all nodes i of the module M_j . If the delay assigned to module M_j is D_j , then the best delay assignment for minimizing power is obtained when

$$\frac{D_1}{C_1} = \frac{D_2}{C_2} = \dots = \frac{D_j}{C_j} \quad (5-1)$$

It is clear that such an assignment of delay to each M_j can cause the overall path delay constraint (sum of delays assigned to each module) to be violated for some of the paths in the module. Therefore, the iterative HA²TSD algorithm is used to solve the problem.

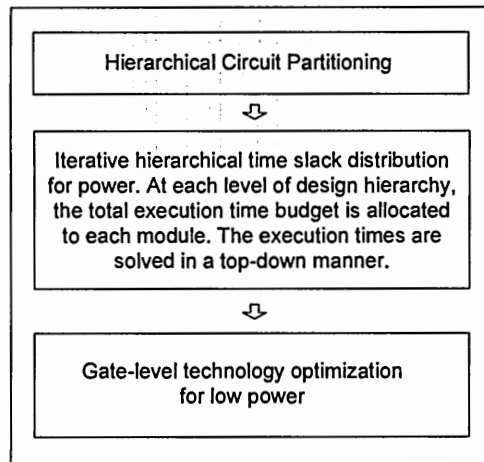


Figure 44. Power optimization overview.

5.2.1 TOPOLOGICAL DEPTH-BASED PARTITIONING

For simulation run-time efficiency and power optimization effectiveness, we introduce a circuit partitioning algorithm which ensures the minimization of the delay skew between sub-modules, and constrains maximum sub-module size (or fan-out size). Figure 45 gives conceptual overview of the topological depth-based partitioning. First of all, labeling of each circuit node is conducted according to the topological order. Then, according to the maximum depth and maximum size constraints, the whole flattened gate-level digital circuit is partitioned into sub-module circuits. The detailed algorithm for the partitioning is shown in Figure 46. The complexity of this algorithm is $O(b^m)$, where b is the branching factor (i.e., average fan-out number) and m is maximum topological depth.

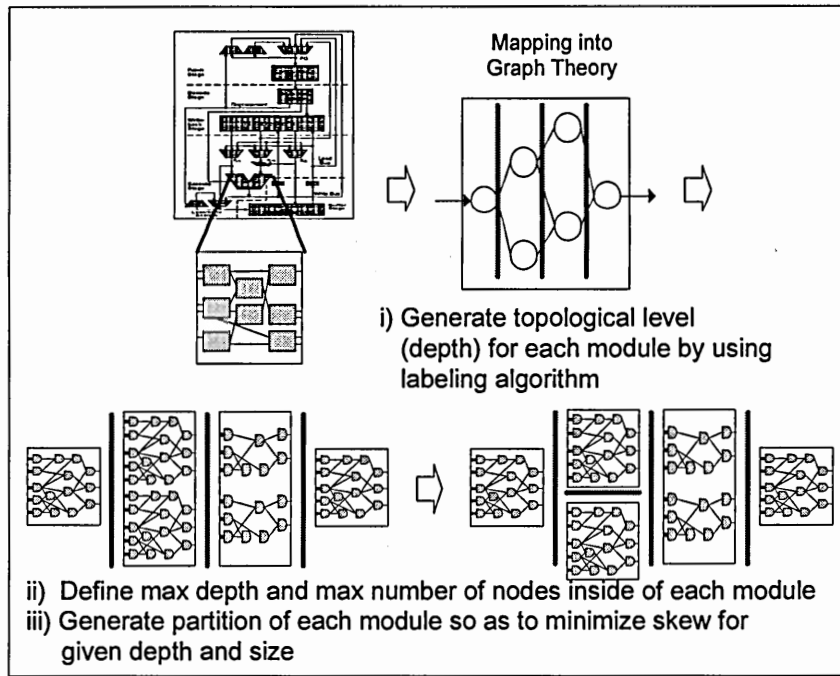


Figure 45. Partitioning overview.

HA2TSD Algorithm 1 : Partitioning

Input: Directed acyclic graph $G = (V, E)$
Output: Partitioned sub-graphs $G_i (V_i, E_i)$, $i = 2 \dots n$

Begin
Phase 0: Initialization
Initialize the class variable of all nodes V in G ;
(level (topological order) = ∞)

Phase I: Labeling (Identify topological order)
Put a start node to FIFO Queue q ;
Initialize the level of the start node = 0;
While (q is not 0)
{
 Obtain a reference to the first element(x) in q ;
 Remove node x from q ;
 For each fan-out node y of node x
 {
 If level of $y = \infty$ **then**
 {
 If number of fan-in node of $y = 1$ **then**
 level of $y =$ level of $x + 1$;
 else if number of fan-in node of $y > 1$ **then**
 level of $y = \text{MAX}(\text{levels of fan-in nodes of } y) + 1$;
 Add node y into q ;
 }
 }
}
Phase II: Partitioning
Sort all nodes according to the topological order;
Partition the graph G by constraints (max node number & depth);
End

Figure 46. Partitioning algorithm.

5.2.2 ACTIVITY-AWARE DELAY ASSIGNMENT

In this section, we illustrate the hierarchical delay assignment algorithm by using an example of the module level circuit. Let's say we have the following functional modules of the target system after partitioning and directed-acyclic-graph mapping as shown in Figure 47.

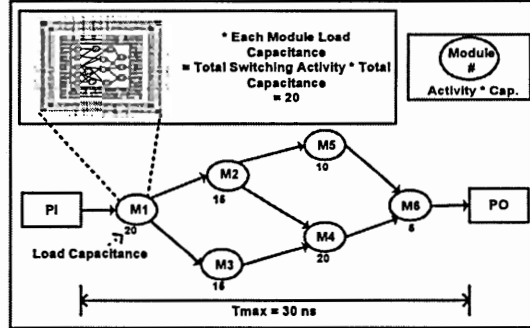


Figure 47. An example module-level circuit.

The object is that given total maximum time (in this case 30 ns), maximum delay of each module is assigned for minimizing total power (summation of the each module's power). In the first step, each module is sorted by the amount of load capacitance of each module (step 1) as shown in Figure 48. Then the highest priority module (the module which has most power-saving possibility) is selected. In this example, we select module *M4* as the highest priority module. After selecting module *M4*, we assign maximum delay for module *M4* by using the "objective function" and "delay assignment" formula as we described at the beginning of the section V. The "objective function" and the "delay assignment" formula is:

$$\text{Object Function} = \frac{D1}{C1} = \frac{D2}{C2} = \dots = \frac{D6}{C6}$$

$$D1 + D2 + D3 + D4 + D5 + D6 \leq T_{\max}$$

$$C_j = \sum_{\text{Node } i} \eta_{ij} c_i \quad \begin{array}{l} C_j : \text{Load Capacitance} \\ \eta_{ij} : \text{Activity} \\ c_i : \text{Capacitance} \end{array}$$

$$\text{Delay assignment} = \frac{C_j}{\text{Total Load Capacitance In the Path}} \times T_{\max}$$

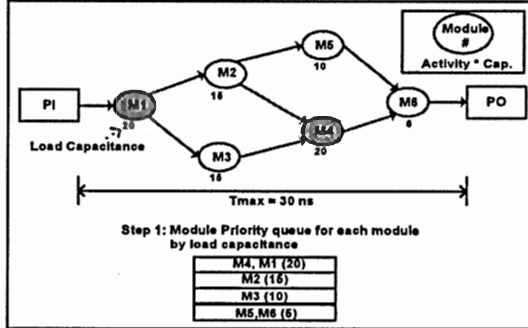


Figure 48. Module-level delay assignment (step 1).

In step 2, all paths which have module *M4* are enumerated according to the total load capacitance of each path and we make another priority queue, which is called priority path queue. Then we select the highest priority path on the queue. In this example, the *path 1* is selected. Therefore, as shown in Figure 49, the delay of module *M4* becomes 20ns ($=20/60 * 30$) from the above “object function” and “delay assignment” formula because the module load capacitance of *M4* (C_{M4}) is 20, total load capacitance on the *path 1* is 60, and the max delay for the path is 30 ns. These step1 and step 2 are repeated until all modules have their maximum delay as shown in Figure 50 (step 3).

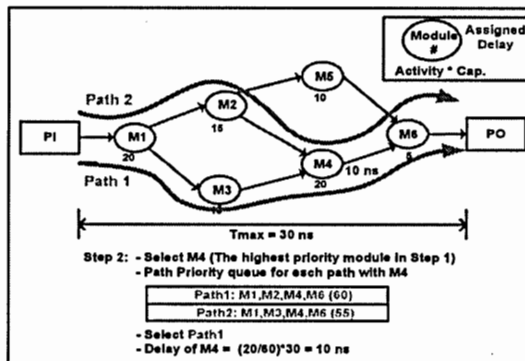


Figure 49. Module-level delay assignment (step 2).

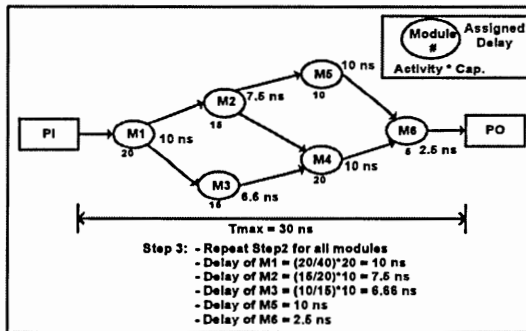


Figure 50. Module-level delay assignment (step 3).

In step 4, we check the criticality of all the paths and if any path is not critical, we conducted local heuristic improvement on the path. The path in Figure 51 is not critical, so we increase the delay of module *M3* from 6.6ns to 7.5ns. Figure 52 shows the final results of the delay assignment for this example module-level circuit. If all modules' delays are assigned and all the paths are satisfied the criticality, conduct the technology parameter optimization at gate level (step 5).

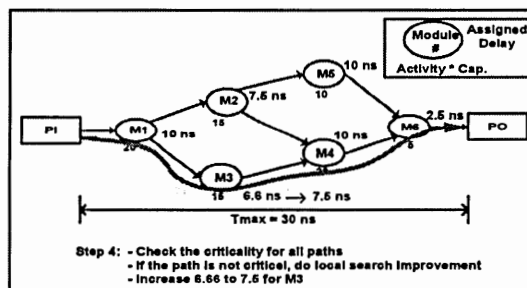


Figure 51. Module-level delay assignment (step 4).

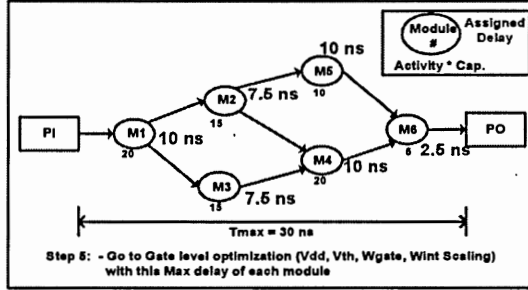


Figure 52. Module-level delay assignment (step 5).

In the algorithm, each module (M_1, \dots, M_i) can be a functional module or a sub-partitioned sub-module, the total physical capacitance of a module is sum of the fan-in/out counts inside the module, and the load capacitance of each module can be calculated by multiplying the total switching activities by the total fan-in/out net counts. Algorithm 2 in Figure 53 illustrates the algorithm. The complexity of the algorithm is $O(nb^m)$, where n is the number of modules, b is the branching factor (i.e., average fan-out number) and m is maximum topological depth.

HA2TSD Algorithm 2 : Delay Assignment

Input: Partitioned sub-graphs $G_i (V_i, E_i)$
Output: Delay weighted sub-graphs $G_i (V_i, E_i, W(v_i))$

Begin

Phase 0: Initialization
Enumerate the critical paths P_j in $G = \{G_1 \dots G_i\}$;
Sort P_j in decreasing order of criticality;

Phase I: Delay assignment for each path
Identify maximum delay T_{max} of all paths;
Calculate switching activity α_i for all nodes V_i
Set the delay for nodes V_i on critical path(s)

While (all path P_j)
{
 While (unassigned $V_i = 0$)
 {
 $T_{max}(V_i) = [\alpha_i / (\alpha_i + \dots + \alpha_{i-1} + \alpha_{i+1} + \dots + \alpha_n)] * T_{max}$;
 /* where n = number of nodes on the P_j */
 $W(V_i) = T_{max}(V_i)$;
 }
}

End

Figure 53. Hierarchical delay assignment algorithm.

5.3 Experimental Results

We developed a simulation frame work with C/C++/STL and Perl on Ultra-80 Unix machine for the hierarchical power optimization. Also, we used off-the-shelf commercial tools for the RTL description, the functional verification, and the logic synthesis of the target system. A few arithmetic modules from the target system and ISCAS89/MCNC91 benchmark circuits are used for the experimental demonstration. For the range of the technology parameter values, the 2001 updated version of ITRS (International Technology Roadmap for Semiconductors) and the MOSIS (Integrated Circuit Fabrication service) parameter test results with TSMC 0.25 micron are used. For the RTL design, we used verilog hardware description, for the functional simulation, we

used VCS (synopsys), and for the logic synthesis, we used design analyzer (synopsys) with 0.25 micron TSMC library. *Monte Carlo simulation* is performed for activity profiling of each module/sub-module as described in [5-2]. This approach consists of applying randomly generated input patterns at the primary inputs of the circuit and monitoring the switching activity per time interval T using a simulator. Under the assumption that the switching activity of a circuit module over any period T has a normal distribution, and for a desired percentage error in the activity estimate and a given confidence level, the number of required simulation vectors is estimated. The simulation based approach is accurate and capable of handling various device models, different circuit design styles, single and multi-phase clocking methodologies, tristate drives, etc. The experimental results of the proposed HA²TSD algorithms in terms of power savings and cpu time reductions are shown in Section 6 with gate-level gate/device-interconnect co-optimization in ultra-deep sub-micron technology.

5.4 Summary

In this chapter, we proposed an efficient approach for minimizing total power (switching, short-circuit, and leakage power) without performance loss for ultra low-power CMOS circuits. We introduce a hierarchical activity-aware delay assignment algorithm, which distributes the surplus time slack into the most power-hungry modules hierarchically. Given the delay constraints, we present a framework for combining supply/threshold voltage scaling, gate sizing, and interconnect scaling for minimizing total power in Section 6. The proposed delay assignment algorithm and the device-

interconnect co-optimization ensures that the total power is minimal in the presence of back end (post-layout-based) UDSM (ultra deep sub-micron) effects.



CHAPTER 6

ULTRA DEEP-SUB-MICRON (UDSM)-AWARE GATE-LEVEL POWER OPTIMIZATION FOR LOW-POWER CMOS VLSI

6.1 Introduction

As technology sizes continue to decrease (with features below 0.1 micron), many new effects are being observed due to the use of ultra-deep sub-micron (UDSM) technologies. The significant UDSM effects are caused by i) *leakage power increase* which is the most undesirable short-channel effect and ii) *on-chip interconnect limits* which potentially threaten to decelerate or halt the historical progression of the semiconductor industry because the miniaturization of interconnects, unlike transistors, does not enhance their performance. For the past two decades the driving force for integrated circuits has been scaling of both the devices and interconnects [6-2~6-5]. We believe that the most effective approach to power optimization in UDSM designs is to

consider both device and interconnect properly throughout the entire design process from RTL level to layout design according to the need of the design complexity and the accuracy.

Layout-based on-chip interconnect effects due to electrical, thermal, and mechanical stresses in a multilevel interconnect stack and MOSFET's short channel effects are considered to minimize the total (switching, short-circuit, and leakage) power. In this chapter, we describe a strategy for solving the following problem for ultra-low power CMOS circuits in nanometer technologies.

Given: i) A logic or gate net-list from synthesizable applications, ii) device/interconnect technology and parasitics from initial layout, iii) a required operational clock frequency, iv) activity profiles at each input, and v) delay/area constraints.

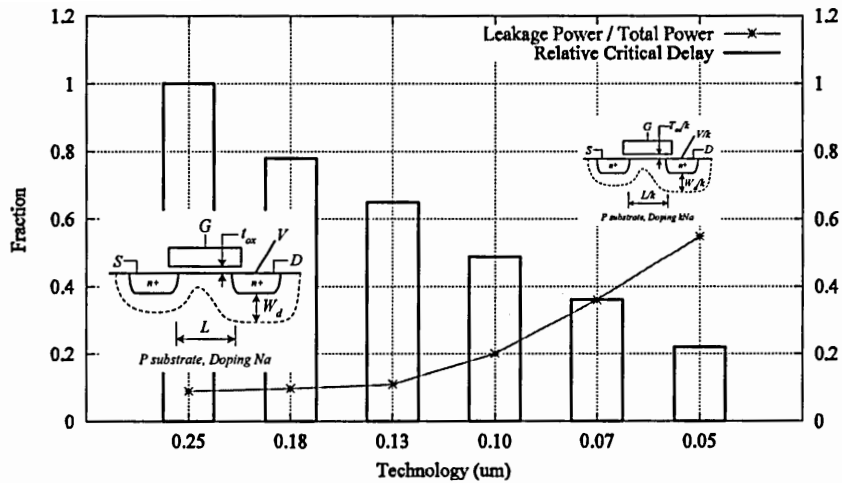
Minimize: $Total_Power(Vdd, Vth, Wgate, Wint)$

Subject to: $Delay(Vdd, Vth, Wgate, Wint) \leq Tspec$
 $Vddi = Vdd-optimal(dual), \forall gate i$
 $Vthi = Vth-optimal(multiple), \forall gate i$
 $Maxsize(i) \geq Wgatei \geq Minsize(i), \forall gate i$
 $Maxsize(j) \geq Wintj \geq Minsize(j), \forall int j$

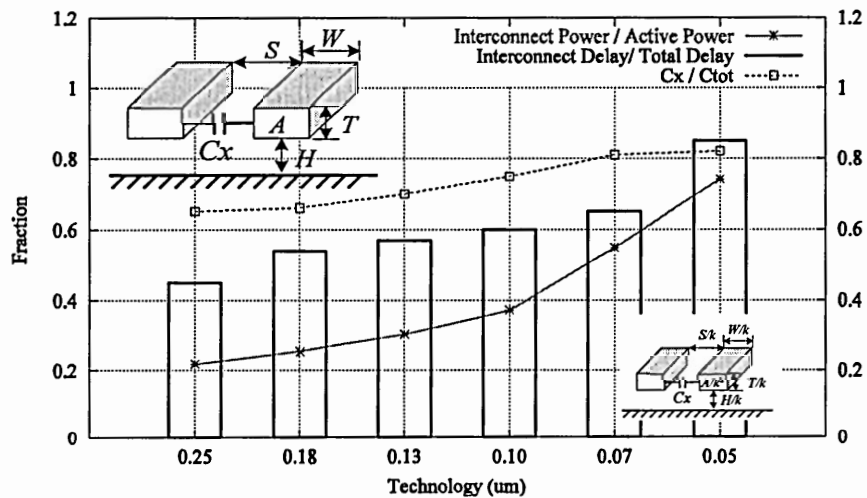
Determine: i) The optimal supply voltage Vdd , ii) the threshold voltage $Vthi$ of each MOSFET, iii) the channel width $Wgatei$ of each MOSFET, and iv) the interconnect width $Wintj$ of each interconnect such that the sum of the static, dynamic and short-circuit components of energy consumption in a clock is minimized while allowing operation at the desired clock frequency fc .

6.2 UDSM-Aware Design

Significant UDSM effects in terms of power and delay are mainly caused by low-threshold voltage and high density-interconnections as shown in Figure 54. We demonstrate those effects with 12-typical benchmark circuits (MCNC91, ISCAS89) by using the parameters in Table 13. As shown in Figure 54(a), leakage power portion is getting larger as the technology feature size is smaller. And the interconnect delay and power have become dominating factor in determining the circuit performance in UDSM designs because of mainly two reasons: i) increased routing densities that have led to shrinking interconnect pitches and ii) aspect ratios that attempt to keep the resistance of these narrower interconnects constant as shown in Figure 54(b). Both have resulted in an increase in the capacitance per unit length, particularly due to interlayer coupling capacitance. Interconnect scaling approaches have been introduced for the optimal interconnect structure of each net in terms of interconnect topology, wire width and spacing, buffer locations and sizes to meet the performance and signal reliability requirements [6-3].



(a) Leakage Impact



(b) Interconnect Impact

Figure 54. UDSM Effects for 12-Banchmark Circuits

Table 13. Summary of Technology Parameters [6-1]

Technology (nm)	250	180	130	100	70	50	
Min. Vdd (v)	2.0	1.8	1.5	1.3	0.9	0.6	
Vth-high (v)	1.0	0.9	0.75	0.65	0.45	0.3	
Vth-low (v)	0.5	0.45	0.375	0.325	0.225	0.1	
Tox (A°)	60	45	35	30	20	12	
Wgate (um)	5.75	4.1	3.0	2.3	2.8	3.9	
fc (GHz)	1.0	1.2	1.6	2.0	2.5	3.0	
Levels	6	6	7	8	9	9	
Poly	H (um)	0.2	0.15	0.13	0.1	0.07	0.07
	W (um)	0.25	0.18	0.13	0.1	0.07	0.05
	S (um)	0.25	0.18	0.13	0.1	0.07	0.05
	R (ohm)	4	5.3	6.2	8	11.4	11.4
M1-2	H (um)	0.5	0.46	0.34	0.26	0.2	0.14
	W (um)	0.30	0.23	0.17	0.13	0.1	0.07
	S (um)	0.30	0.23	0.17	0.13	0.1	0.07
	T (nm)	650	500	360	320	270	210
	R (ohm)	0.044	0.048	0.065	0.085	0.11	0.16
M3-4	H (um)	2.0	2.0	1.2	1.0	0.6	0.6
	W (um)	1.0	1.0	0.6	0.5	0.3	0.3
	S (um)	1.0	1.0	0.6	0.5	0.3	0.3
	T (nm)	900	900	900	900	900	900
	R (ohm)	0.011	0.011	0.018	0.0224	0.036	0.036
M5-6	H (um)	2.5	2.5	2.0	2.0	1.5	1.5
	W (um)	2.0	2.0	1.0	1.0	0.75	0.75
	S (um)	2.0	2.0	1.0	1.0	0.75	0.75
	T (nm)	1400	1400	900	900	900	900
	R (ohm)	0.009	0.009	0.011	0.011	0.015	0.015
M7-8	H (um)	-	-	2.5	2.5	2.4	2.4
	W (um)	-	-	2.0	2.0	1.2	1.2
	S (um)	-	-	2.0	2.0	1.2	1.2
	T (nm)	-	-	1400	1400	900	900
	R (ohm)	-	-	0.009	0.009	0.0094	0.0094
M9	H (um)	-	-	-	-	2.5	2.5
	W (um)	-	-	-	-	2.0	2.0
	S (um)	-	-	-	-	2.0	2.0
	T (nm)	-	-	-	-	1400	1400
	R (ohm)	-	-	-	-	0.009	0.009
Via (M1-M2)	size (um)	0.5	0.36	0.26	0.2	0.14	0.1
	R (ohm)	0.46	0.69	0.95	1.43	2.16	3.27
k		3.3	2.7	2.3	2.0	1.8	2.5

Performance-driven logic synthesis followed by performance-driven layout [6-4,6-5,6-9] has become a necessity for designing high performance circuits. However, this loosely coupled two-phase timing optimization methodology has serious limitations for deep-submicron-based designs in which interconnection delays become the dominant factor in determining the circuit speed. Accurate information regarding the

interconnection delays is not available during the logic synthesis phase and the interconnection delays are just roughly estimated in this phase. The estimation errors in the estimation could result in a logic design far-off from an optimal one. Therefore, in this research, we use post-layout parameters for interconnect scaling.

6.3 Delay and energy model

We used reasonably accurate UDSM-aware model (below 10% errors compared to HSPICE level 49 model) for delay and power, compared with most of the literature [6-15~6-17], of MOSFET itself and also interconnect component of CMOS circuits.

6.3.1 MOSFETs IN UDSM

Since Shockley's square model is not accurate for UDSM's MOSFETs due to velocity saturation, short channel effect, mobility degradation, and series resistance, new MOSFET's models have been developed to replace it [6-15]. Traditional long-channel MOS theory states that device current in the saturation mode of operation is proportional to the square of gate drive ($V_{dd}-V_{th}$) and inversely proportional to channel length. At ultra-short channel lengths most carriers travel at a maximum saturated velocity, V_{sat} , throughout the channel, which nearly eliminates the impact of channel length on current. We used a recent version of alpha-power law model [6-15~6-17].

6.3.2 INTERCONNECTS IN UDSM

In UDSM, interconnect models must be carefully monitored and controlled since the impact of interconnect on system performance is rising. Physically meaningful voltage is developed only across a capacitance and the interconnect capacitance in UDSM dominates the total net capacitance due to; i) the increased routing densities that have led to shrinking wiring pitches and ii) the aspect ratios that attempt to keep the resistance of these narrower wires constant. Many researches have been studied for modeling of the coupling effect in time domain [6-16] and also frequency domain. In general, frequency domain approaches are ideal for measurement of high-frequency phenomena such as skin effect but unsuitable for recreating the actual environment in real designs [6-17]. We used post-layout time-domain model for the interconnect effect so that the analysis results give easily measurable delay and energy, rather than hard-to-measure frequency response and phases.

6.3.3 COMPONENT DELAY (GATE+INTERCONNECT) MODEL

We use a transregional model for estimating the worst-case signal propagation delay through a gate. The delay model has been derived using an extension of the alpha-power law saturation drain current model to the subthreshold region. The drain current model incorporates effects of high-field and quasi-ballistic (velocity overshoot) carrier transport in the MOSFET channel. The delay model consists of four major components: i) the delay due to switching MOSFETs, ii) the distributed interconnect RC delay, iii) the time of flight delay, iv) the delay component due to the non-zero rise time of the input

signal are considered. These definitions of gate delay and interconnect resistance delay allow the definition of arrival times and required times at the input and output of a gate in the network, which are used for defining time slack.

6.3.4 COMPONENT ENERGY (GATE+INTERCONNECT) MODEL

The equations used to compute the dynamic and static energy dissipations of a gate with similar models which have been presented and analyzed in a recent work from original work in [6-15]. It is assumed that the gates are simple multi-input gates with symmetric series or parallel pull-up and pull-down MOSFET configurations. Contributions of subthreshold leakage through the MOSFET channel as well as the leakage across the device drain junctions to static dissipation are included.

6.4 UDSM-Aware Post-Layout Power Optimization

We believe that the most effective approach to power optimization in UDSM designs is to consider both device and interconnect designs at gate level after initial layout in terms of optimization complexity, simulation efficiency, and estimation accuracy. The proposed design flow of our approach is shown in Figure 55.

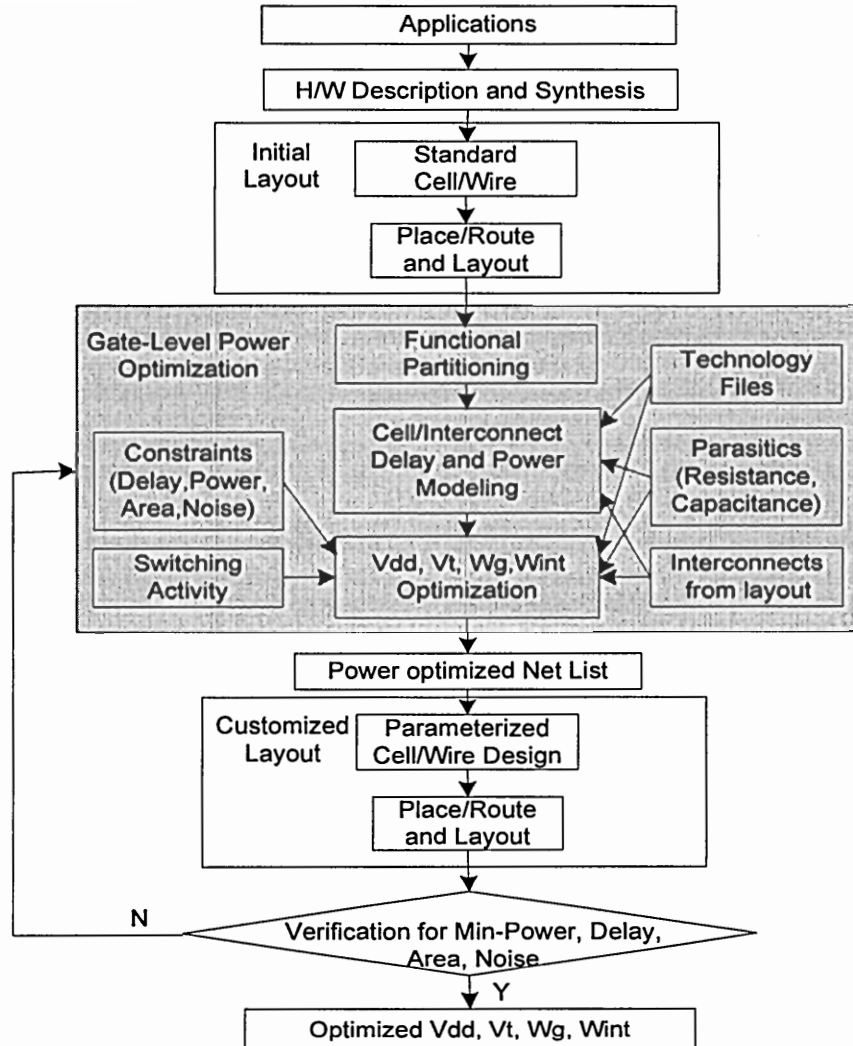


Figure 55. Proposed power optimization design flow.

At the highest level, hierarchical design decomposition is first performed via architectural partitioning. Then, using simulation based methods, activity metrics for all the architectural modules and sub-modules are derived. Next, system-level timing constraints are decomposed into module-level delay specifications. This module-level delay decomposition is based on an energy-delay-ratio (EDR) heuristic that prescribes how module-level delays can be computed from system delay (timing) requirements in

such a way as to minimize total power consumption. The EDR heuristic uses knowledge of the effective capacitance of each module in the hierarchical design description to perform near-optimal (in terms of power) delay assignment. The procedure is applied hierarchically to every system module while delay values for gate/device-interconnect level modules are obtained.

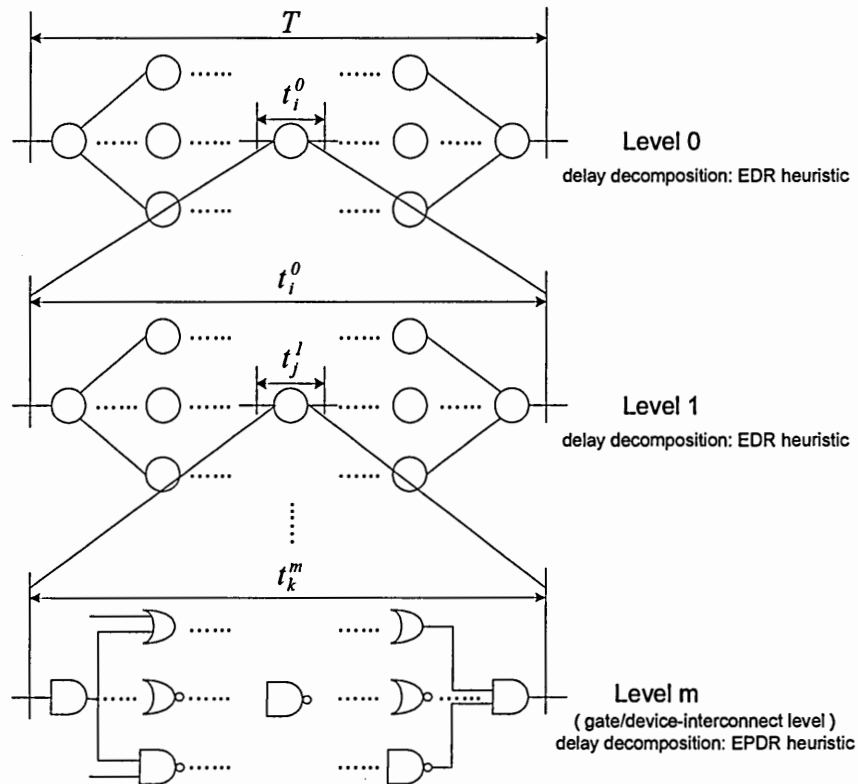


Figure 56. Hierarchical delay decomposition.

In Figure 56, LEVEL 0 represents the highest level of design description. The system-level delay requirement T^0 in the Figure 56 is decomposed into delay requirements for the i th module at level 0 using the EDR heuristic. The procedure is repeated

hierarchically obtaining the delay values of the j th module and level of h of the design hierarchy. At the lowest level (level m in Figure 56) each module consists of logic gates and associated interconnect. At this level of design description, the individual paths in the logic module can be determined efficiently and a different path-based delay assignment heuristic called the EPDR (energy*path effort and delay ratio) heuristic is used to optimize the logic gates and interconnect for minimum power. This is because at the lowest level of design hierarchy, accurate path-based delay analysis of the logic gates and interconnect can be performed taking into account UDSM scaling effects. At higher levels of design hierarchy, it is sufficient to perform worst-case delay assignment for all the logic modules using EDR metric.

6.4.1 PRE-PROCEDURE

For simulation run-time efficiency and power optimization effectiveness, we used our topological depth-based partitioning [6-7] which ensures the minimization of the delay skew between sub-modules, and constrains maximum sub-module size (or fan-out size). First of all, labeling of each circuit node is conducted according to the topological order. Then, according to the maximum depth and maximum size constraints, the whole flattened gate-level digital circuit is partitioned into sub-module circuits. The complexity of this algorithm is $O(b^m)$, where b is the branching factor (i.e., average fan-out number) and m is maximum topological depth.

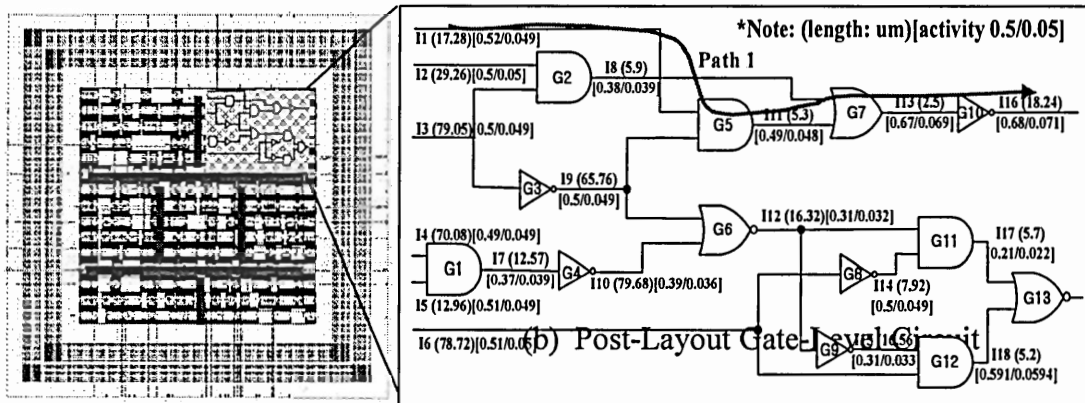
We used *Monte Carlo simulation* for activity profiling of each module/sub-module. This simulation based approach is accurate and capable of handling various

device models, different circuit design styles, but take very long simulation time. However, our hierarchical partitioning approach can be used for compensating this problem.

We placed and routed the synthesized circuits by using feasible technology library and then scaled the device/interconnect dimensions such as oxide thickness, effective channel length/width, and interconnect length /width according to the scaling factor in [6-1,6-17]. For some geometric and electrical parameters like interconnect capacitance/resistance, W/S/T/H of each metal layer, Vdd/Vth range of each technology, etc..., we followed 2001 edition of the ITRS projections [6-1].

After obtaining the post-layout gate-level net lists, we enumerated all possible paths for the circuits.

Figure 57 shows the pre-procedure for a benchmark circuit.



- Path 1: I(1)-G(5)-I(11)-G(7)-I(13)-G(10)-I(16)
- Path 2: I(2)-G(2)-I(8)-G(7)-I(13)-G(10)-I(16)
- Path 3: I(3)-G(2)-I(8)-G(7)-I(13)-G(10)-I(16)
- Path 4: I(3)-G(3)-I(9)-G(5)-I(11)-G(7)-I(13)-G(10)-I(16)
- Path 5: I(3)-G(3)-I(9)-G(6)-I(12)-G(11)-I(17)-G(13)-I(19)
- Path 6: I(3)-G(3)-I(9)-G(6)-I(12)-G(9)-I(15)-G(12)-I(18)-G(13)-I(19)
- Path 7: I(4)-G(1)-I(7)-G(4)-I(10)-G(6)-I(12)-G(11)-I(17)-G(13)-I(19)
- Path 8: I(4)-G(1)-I(7)-G(4)-I(10)-G(6)-I(12)-G(9)-I(15)-G(12)-I(18)-G(13)-I(19)
- Path 9: I(5)-G(1)-I(7)-G(4)-I(10)-G(6)-I(12)-G(11)-I(17)-G(13)-I(19)
- Path 10: I(5)-G(1)-I(7)-G(4)-I(10)-G(6)-I(12)-G(9)-I(15)-G(12)-I(18)-G(13)-I(19)
- Path 11: I(6)-G(8)-I(14)-G(11)-I(17)-G(13)-I(19)
- Path 12: I(6)-G(12)-I(18)-G(13)-I(19)

(c) Path Enumeration

Figure 57. Pre-procedure for c499.

6.4.2 PROPOSED HEURISTIC

In general, there are several methods to solve non-linear optimization problem for VLSI circuit optimization; i) exhaustive search, ii) restriction (work on subset of the original problem), iii) pseudo-polynomial time algorithm like Lagrangian and Dynamic Programming, iv) approximation or randomization, and v) heuristics. Our experience shows that heuristic method is the best fit for this device and interconnect co-optimization problem (The experimental validation is not included in this research due to

page limit). For better understanding of our proposed algorithm, we illustrate our algorithm with the example circuit in Figure 57(b). The Figure 58 shows that power and the delay for the example circuit of Figure 57(b) as a basis before optimization with 0.05 um technology (see Table 13 for the parameters used)

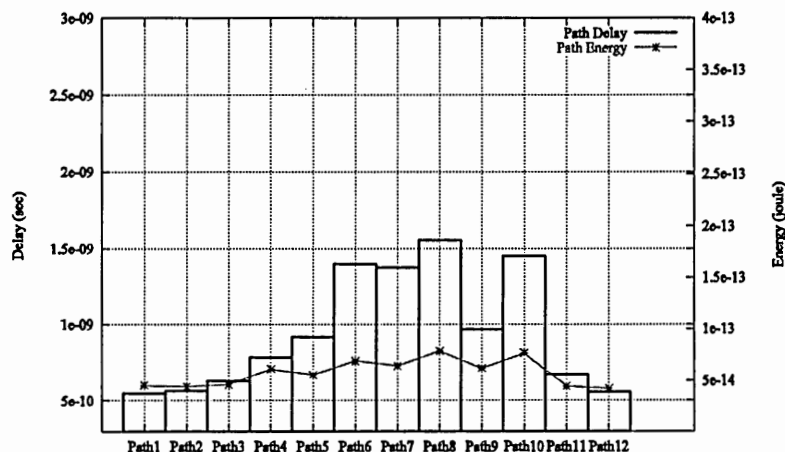


Figure 58. Power and delay basis for an example circuit in Figure 57 (b) without optimization.

Step 1: Set all parameters for minimum power. In this phase, all the paths show slowest delay within the technology parameter ranges and violate the delay constraints. The Figure 59 shows the power and the delay for the above example circuit.

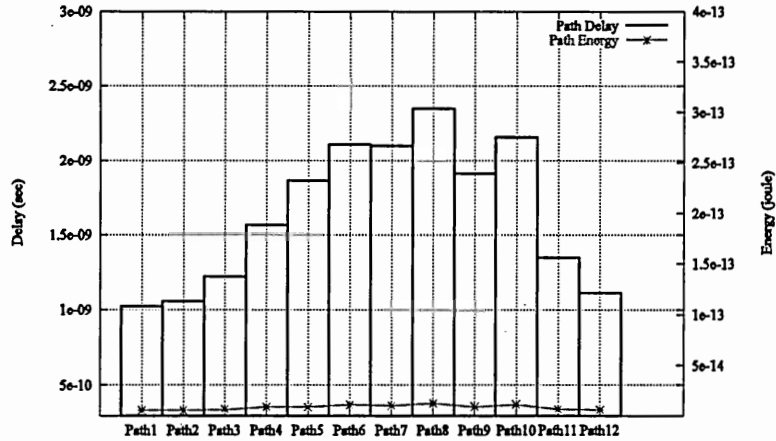


Figure 59. Technology mapping for minimum power.

Step 2: Select delay violation paths according to delay constraints. Path 5, 6,7,8,9,10 are the violation paths in this example as shown in Figure 60.

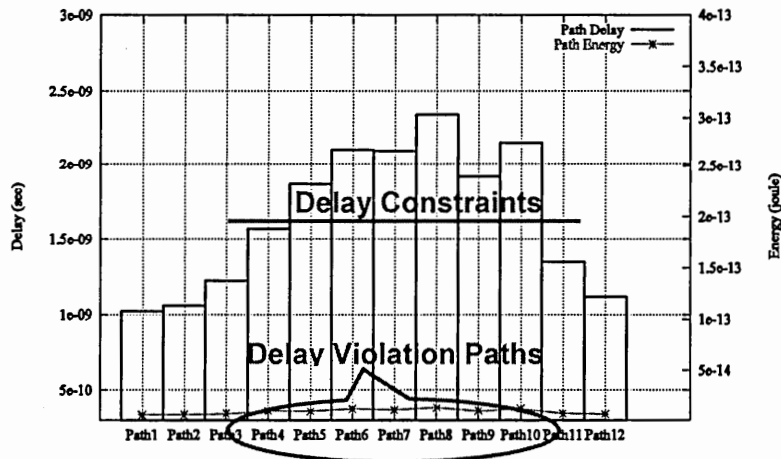


Figure 60. Violation paths.

Step 3: Determine the amount of delay to be decreased for each component on the violation paths by using EPDR (Energy*Path Effort and Delay Ratio) paradigm for minimizing power.

Lemma 6-1: EPDR (Energy*Path-effort and Delay Ratio) Paradigm: When the ratios of the (*component energy*path effort*) and the *component delay* for each module are same, the total energy consumption is minimal. Therefore the surplus time slacks should be assigned onto each component according to the cost function of $\frac{E_1 \cdot P_1}{d_1} = \frac{E_2 \cdot P_2}{d_2} = \dots = \frac{E_n \cdot P_n}{d_n}$

(1,..., n: component number) for minimum power.

The rationale of the EPDR algorithm is that the component which has more violation paths and less energy consumption possibility due to the delay reduction should have higher priority to reduce some portion of the amount of the violation delay than others.

Step 4: Find Vdd, vth, Wg, Wint of each component to minimize the increase of the power for each component on the violated paths when the delay of each component is reduced up to the amount of the delay from Step 3. The cost function to be minimized is

$$\frac{\Delta Power(Vdd, Vth, Wg, Wint, \eta)}{\Delta Delay(Vdd, Vth, Wg, Wint)}$$

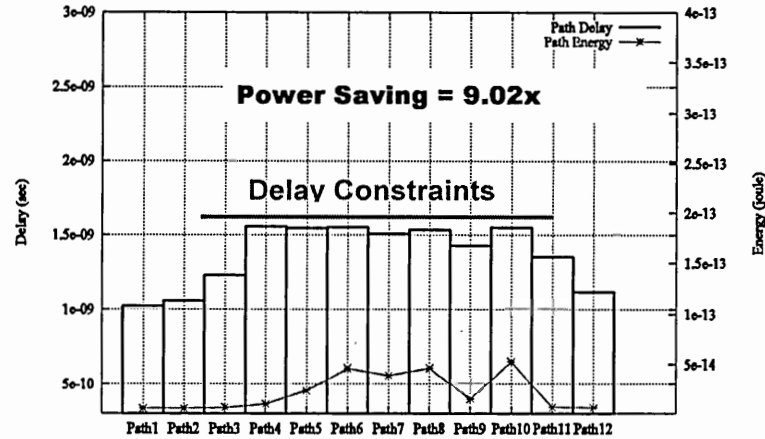


Figure 61. After optimization.

Step 5: Repeat the Step 2-4 until all paths satisfy the delay bound. Figure 61 shows proposed power optimization result for the example circuit.

6.5 Experimental results

We developed a simulation frame work with C/C++/STL and Perl on Ultra-80 Unix machine for the hierarchical power optimization. Also, we used off-the-shelf commercial tools (synopsis and cadence) for the RTL description, the functional verification, and the logic synthesis / initial layout of the target system. A few arithmetic modules from ARM Core and ISCAS89/MCNC91 benchmark circuits are used for the experimental demonstration. For the range of the technology parameter values, the 2001 updated version of ITRS (International Technology Roadmap for Semiconductors), the MOSIS (Integrated Circuit Fabrication service) parameter test results with our scaling

techniques which is proven by HSPICE.

Table 14 shows the total power consumption with fixed threshold voltage with 0.05 micrometer technology for the given circuits. Table 15 demonstrates the effectiveness of the flattened case power optimization. The experimental results show that our power optimization strategy delivers average 2x – 15x savings in total (static and dynamic) power without performance degradation. Table 16 (level of hierarchy is 2) shows average 35.08% faster optimization than the totally flattened case when we still have average 4.72x power savings. With the hierarchical depth of 3 and 4 as shown in Table 17 and Table 18, we can obtain average 46.10%-50.21% faster optimization than the totally flattened case when we still have average 3.31x-2.89x power savings. Figure 62 and Figure 63 show that the average power savings and the cpu time savings when we use different size of partitioning. Figure 64 and Figure 65 show that the average power savings and the cpu time savings when we use different level of hierarchy. Figure 66 shows the comparison with exhaustive optimal solution, randomly averaged case, and our proposed technique for a benchmark circuit (s27). It shows that our algorithm performs reasonable well and fast. In addition, our experiments show that judiciously chosen partitioning also can significantly help to reduce the time complexity, maintaining similar power optimization results.

Table 14. Before Optimization

BEFORE OPTIMIZATION (TECHNOLOGY: 0.05 MICROMETER, FIXED VTH=0.7, OPTIMAL VDD=0.6-1.2, OPTIMAL W=1-100)

System Module	Gates/Depth	Delay(ns)	Input Activity	Vdd (v)	W_min	W_avg	W_max	Static	Dynamic	Short-ckt	Total
4-Full Adder	106/48	3.36	0.5	0.725	2	15	24	2.19E-15	1.45E-11	2.22E-12	1.67E-11
			0.05	0.750	2	15	20	2.19E-15	1.33E-13	2.95E-14	1.65E-13
16-Lookahead	1838/81	7.00	0.5	0.725	2	19	56	8.02E-15	4.95E-11	2.26E-12	5.18E-11
			0.05	0.725	2	18	54	8.02E-15	1.98E-13	3.86E-14	2.45E-13
64-ALU	3417/226	13.60	0.5	0.675	2	23	68	1.12E-14	4.40E-10	2.87E-11	4.69E-10
			0.05	0.675	2	20	62	1.12E-14	1.90E-11	2.87E-13	1.93E-11
s298	286/18	3.02	0.5	0.725	2	16	24	1.92E-15	1.44E-12	2.37E-14	1.47E-12
			0.05	0.750	2	16	20	1.92E-15	1.39E-14	2.55E-16	1.61E-14
s344	229/28	3.86	0.5	0.725	6	16	34	4.59E-16	6.38E-12	9.87E-14	6.48E-12
			0.05	0.750	4	18	36	4.59E-16	6.39E-14	9.62E-16	6.53E-14
s386	426/23	3.99	0.5	0.725	2	18	52	5.56E-15	4.88E-12	9.99E-14	4.99E-12
			0.05	0.725	2	18	46	5.56E-15	5.13E-14	9.62E-16	5.78E-14
s526	596/18	4.30	0.5	0.725	2	26	46	5.88E-15	5.13E-12	2.00E-13	5.34E-12
			0.05	0.750	2	22	68	5.88E-15	5.32E-14	9.82E-16	6.00E-14
c6288	2406/129	10.60	0.5	0.700	2	17	47	6.52E-14	3.21E-10	6.55E-11	3.87E-10
			0.05	0.700	2	19	34	6.52E-14	4.69E-11	6.55E-13	4.76E-11

Table 15. After Optimization

AFTER OPTIMIZATION (TECHNOLOGY: 0.05 MICROMETER, FLATTENED CASE, OPTIMAL VTH=0.1-0.7, OPTIMAL VDD=0.6-1.2, OPTIMAL W=1-100)

System Module	Hierarchy	Delay(ns)	Input Activity	Vdd (v)	Vth (v)	W_min	W_avg	W_max	Total	Power Savings (x)	Run Time (sec)
4-Full Adder	flattened	3.36	0.5	0.625	0.200	2	6	10	6.48E-12	2.58	291.56
			0.05	0.625	0.200	2	8	12	7.57E-14	2.18	245.78
16-Lookahead	flattened	7.00	0.5	0.600	0.200	2	14	34	5.15E-12	10.05	1199.45
			0.05	0.600	0.200	2	13	24	3.14E-14	7.79	998.32
64-ALU	flattened	13.60	0.5	0.675	0.175	2	15	46	3.07E-11	15.27	10872.42
			0.05	0.675	0.175	2	15	45	1.93E-12	10.00	10987.25
s298	flattened	3.02	0.5	0.625	0.200	2	8	14	3.48E-13	4.21	158.32
			0.05	0.600	0.200	2	8	12	3.76E-15	4.28	102.57
s344	flattened	3.86	0.5	0.625	0.200	2	9	24	9.48E-13	6.83	129.53
			0.05	0.625	0.200	2	10	20	1.57E-14	4.16	119.42
s386	flattened	3.99	0.5	0.600	0.175	2	12	34	1.48E-12	3.37	299.41
			0.05	0.600	0.175	2	10	24	1.23E-14	4.70	300.55
s526	flattened	4.30	0.5	0.650	0.200	2	14	32	1.92E-12	2.78	180.42
			0.05	0.625	0.175	2	14	36	2.31E-14	2.60	194.56
c6288	flattened	10.60	0.5	0.650	0.200	2	11	24	4.17E-11	9.28	3875.81
			0.05	0.650	0.200	2	10	18	6.69E-12	7.12	3591.33
Average										6.08	

Table 16. After Optimization (cont'd)AFTER OPTIMIZATION (TECHNOLOGY: 0.05 MICROMETER, 2-LEVEL HIERARCHY CASE, OPTIMAL V_{TH}=0.1-0.7, OPTIMAL V_{DD}=0.6-1.2, OPTIMAL W=1-100)

System Module	Hierarchy	Partitioning	Input Activity	Vdd (v)	Vth (v)	W_avg	Total	Power Savings (x)	Run Time (sec)	Time Savings (%)
4-Full Adder	2	n/3	0.5	0.625	0.200	10	7.58E-12	2.21	155.67	46.61
			0.05	0.625	0.200	11	8.10E-14	2.04	166.97	32.07
16-Lookahead	2	n/3	0.5	0.600	0.175	16	6.82E-12	7.60	765.43	36.18
			0.05	0.600	0.175	15	3.94E-14	6.21	745.38	25.34
64-ALU	2	n/3	0.5	0.675	0.150	20	4.25E-11	11.03	7222.54	33.57
			0.05	0.675	0.150	20	2.30E-12	8.39	7967.65	27.48
s298	2	n/3	0.5	0.625	0.200	13	4.89E-13	3.00	99.32	37.27
			0.05	0.600	0.200	15	4.59E-15	3.50	69.17	32.56
s344	2	n/3	0.5	0.625	0.200	13	1.10E-12	5.89	81.53	37.06
			0.05	0.625	0.200	13	2.84E-14	2.30	75.99	36.37
s386	2	n/3	0.5	0.600	0.175	15	2.18E-12	2.29	168.81	43.62
			0.05	0.600	0.175	14	1.99E-14	2.90	171.25	43.02
s526	2	n/3	0.5	0.650	0.200	17	2.51E-12	2.13	98.22	45.56
			0.05	0.625	0.175	17	2.71E-14	2.21	124.96	35.77
c6288	2	n/3	0.5	0.650	0.200	15	5.34E-11	7.25	2876.85	25.77
			0.05	0.650	0.200	16	7.27E-12	6.55	2765.63	22.99
Average								4.72		35.08

Table 17. After Optimization (cont'd)AFTER OPTIMIZATION (TECHNOLOGY: 0.05 MICROMETER, 3-LEVEL HIERARCHY CASE, OPTIMAL V_{TH}=0.1-0.7, OPTIMAL V_{DD}=0.6-1.2, OPTIMAL W=1-100)

System Module	Hierarchy	Partitioning	Input Activity	Vdd (v)	Vth (v)	W_avg	Total	Power Savings (x)	Run Time (sec)	Time Savings (%)
4-Full Adder	3	n/3	0.5	0.625	0.200	14	8.23E-12	2.03	120.56	58.65
			0.05	0.625	0.200	14	8.01E-14	2.06	129.45	47.33
16-Lookahead	3	n/3	0.5	0.600	0.175	17	9.58E-12	5.40	700	41.64
			0.05	0.600	0.175	17	5.32E-14	4.59	680.32	31.85
64-ALU	3	n/3	0.5	0.675	0.150	20	8.05E-11	5.82	6209.23	42.89
			0.05	0.675	0.125	20	4.00E-12	4.82	6029.32	45.12
s298	3	n/3	0.5	0.625	0.175	14	5.10E-13	2.87	84.12	46.87
			0.05	0.600	0.150	14	4.89E-15	3.29	60.32	41.19
s386	3	n/3	0.5	0.600	0.150	15	1.56E-12	3.20	133.23	55.50
			0.05	0.600	0.150	14	2.06E-14	2.81	130.48	56.59
s526	3	n/3	0.5	0.650	0.200	23	2.56E-12	2.09	89.32	50.49
			0.05	0.625	0.175	22	3.32E-14	1.81	103.43	46.84
c6288	3	n/3	0.5	0.650	0.175	17	9.56E-11	4.05	2035.45	47.48
			0.05	0.650	0.175	17	1.73E-11	2.76	2120.34	40.96
Average								3.31		46.10

Table 18. After Optimization (cont'd)

AFTER OPTIMIZATION (TECHNOLOGY: 0.05 MICROMETER, 4-LEVEL HIERARCHY CASE, OPTIMAL V_{TH}=0.1-0.7, OPTIMAL V_{DD}=0.6-1.2, OPTIMAL W=1-100)

System Module	Hierarchy	Partitioning	Input Activity	V _{dd} (v)	V _{th} (v)	W _{avg}	Total	Power Savings (x)	Run Time (sec)	Time Savings (%)
4-Full Adder	4	n/3	0.5	0.625	0.200	14	9.93E-12	1.68	100.5	65.53
			0.05	0.625	0.200	14	8.73E-14	1.89	111.45	54.65
16-Lookahead	4	n/3	0.5	0.600	0.175	17	9.98E-12	5.19	689.24	42.54
			0.05	0.600	0.175	17	6.78E-14	3.61	656.32	34.26
64-ALU	4	n/3	0.5	0.675	0.150	20	9.12E-11	5.14	6002.25	44.79
			0.05	0.675	0.125	20	6.38E-12	3.02	6009.32	45.31
s298	4	n/3	0.5	0.625	0.175	14	6.01E-13	2.44	67.25	57.52
			0.05	0.600	0.150	14	5.89E-15	2.73	59.32	42.17
s344	4	n/3	0.5	0.625	0.175	13	2.00E-12	3.24	70.03	45.94
			0.05	0.625	0.150	13	3.16E-14	2.07	62.88	47.35
s386	4	n/3	0.5	0.600	0.150	15	1.65E-12	3.03	125.23	58.17
			0.05	0.600	0.150	14	2.23E-14	2.60	129.89	56.78
s526	4	n/3	0.5	0.650	0.200	23	2.79E-12	1.92	82.92	54.04
			0.05	0.625	0.175	22	3.76E-14	1.60	93.33	52.03
c6288	4	n/3	0.5	0.650	0.175	17	9.87E-11	3.92	1875.25	51.62
			0.05	0.650	0.175	17	2.27E-11	2.10	1769.33	50.73
Average								2.89		50.21

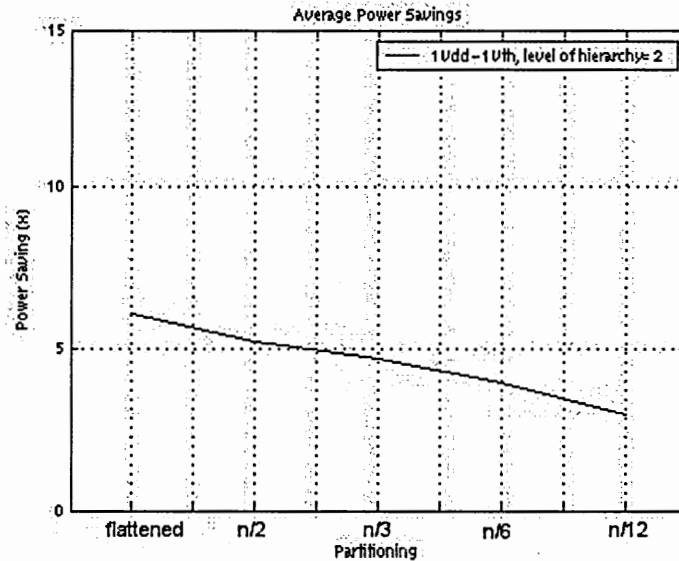


Figure 62. Average power savings in different partitioning.

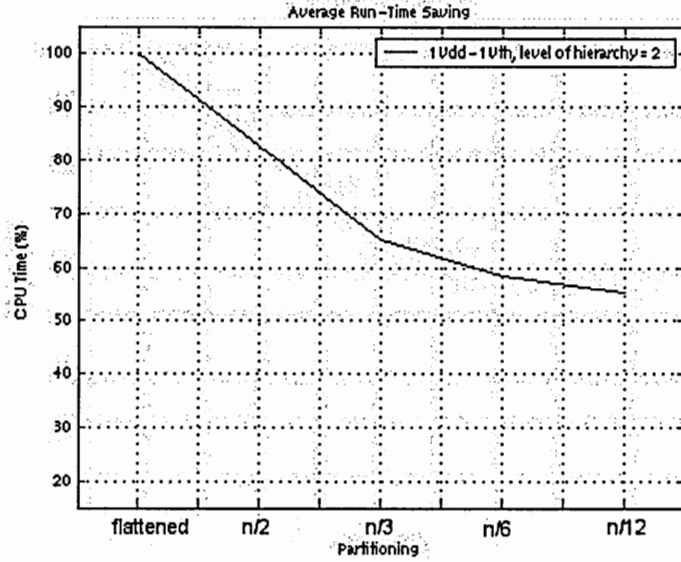


Figure 63. Average time savings in different partitioning.

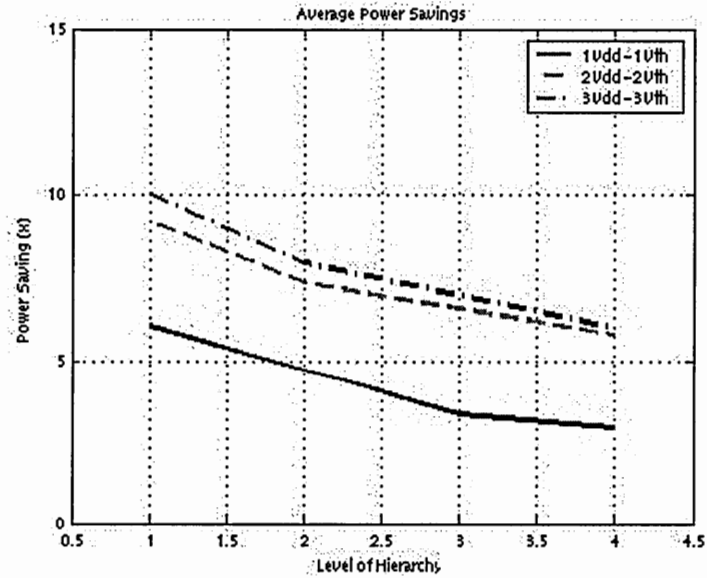


Figure 64. Average power savings in different hierarchies.

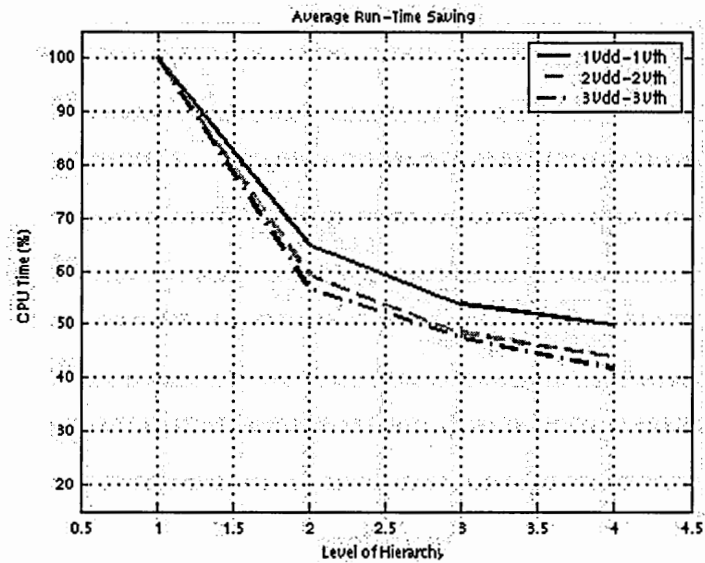


Figure 65. Average time savings in different hierarchies.

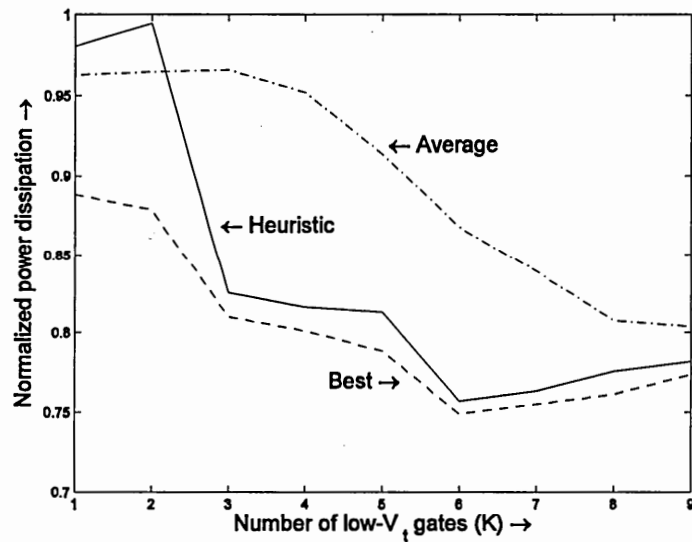


Figure 66. Relative performance.

6.6 Summary

In this chapter, we proposed and implemented an efficient approach to minimize total power (switching, short-circuit, and leakage power) tremendously without performance loss for ultra-low power CMOS circuits in nanometer technologies. We present a framework for combining supply/threshold voltage scaling, gate sizing, and interconnect scaling techniques for power optimization and propose an efficient heuristic algorithm which ensures that the total slack is maximal and the total power is minimal, including UDSM effects and back end (post-layout-based) design techniques. We have tested the proposed algorithms on a set of benchmark circuits and some building blocks of synthesizable ARM core. Consequently experimental results demonstrate that the proposed power optimization strategy is fast and effective for reducing total power in nanometer (below 0.1 micron) technologies, maintaining the circuit speed. Future work will include application-specific and architecture-driven issues with this UDSM-aware post-layout technology mapping techniques because assorted software, algorithmic, architectural techniques can significantly help to minimize the total switching activity for a target system.

CHAPTER 7

CONCLUSION

In this dissertation, we consider the relevant issues involved in low-power design for future generation systems. First, we survey the technology trends and the state-of-the-art techniques for ultra-low power design at each design hierarchy. The literature review shows that the trade-off solutions between the conflicting design criteria (Delay, Area, and Power) should be considered for future low-power, high-performance, and small-size systems. In the research contribution sections, we addressed a set of frame works for software-level, module-level, and technology-level approaches for low power. This chapter concludes the dissertation by first reviewing the salient contributions as well as the key insights and recommendations. Following a summary of the research findings and contributions in section 7.1, the open problems for future investigation are presented in section 7.2 of some possible extensions of this dissertation that could develop into future research endeavors.

7.1 Research Findings

The objective of this dissertation is to develop a vertically integrated framework for software-hardware-technology driven optimization for ultra low-power design of digital systems. To accomplish this research goal, the following contributions have been

identified providing a number of efficient algorithms: i) an efficient instruction –level power optimization, ii) power-ware zero slack algorithm, iii) module-level hierarchical delay assignment approach for minimum power, and iv) deep-submicron-aware gate-level technology optimization scheme.

First, we have developed a framework for describing and analyzing instruction scheduling for low power. This frame work has the following components:

- The instruction scheduling or reordering problem is mapped to the Precedence Constrained Hamiltonian Path problem for DAGs and the Traveling Salesman Problem in general, both of which are *NP-Hard*.
- The proposed instruction scheduling algorithm solves the *NP-Hard* problem in polynomial time by using combining a simulated annealing and a local heuristic.
- Mathematical validation method for the optimization is presented.

The second contribution of this dissertation is a set of algorithms for applying the slack distribution concepts to low-power design. The salient features of these algorithms are listed below:

- A new approach to the problem of slack budget distribution for low power is presented.
- The PA-ZSA ensures that the total slack budget is maximal and the total power is minimal as a power-aware version of the well-known *zero-slack algorithm (ZSA)*.

- To reduce the complexity of the problem, a simple but optimal metric which is called energy-and-delay-ratio (EDR)-based slack distribution algorithm is proposed.
- The experimental results show that our polynomial-time solvable strategy delivers an order of magnitude savings in total (static and dynamic) power over the design without power optimization and our slack distribution strategy reduces more power (average 20% of the total power) over the conventional slack budget algorithms with the same power optimization.

The third contribution of this dissertation is the hierarchical module-level approaches for assigning the module's delays for minimum power. Its characteristics are enumerated next.

- The proposed heuristic distributes the surplus delay into the most power-hungry modules hierarchically.
- Topological depth-based circuit partitioning is presented.
- With the hierarchical approach, polynomial time optimization is feasible in very large circuits.

The final contribution is the demonstration of a new design approach for gate-level low-power design in ultra-deep-submicron technologies. Its features are summarized below.

- UDMSM effects are investigated in terms of power and delay.

- A framework for combining supply/threshold voltage scaling, gate sizing, and interconnect scaling techniques is presented including UDSM effects and back end (post-layout-based) design techniques.
- Path effort (branch effort) feature for multiple paths is added to the delay optimization technique (the EDR paradigm) for a single path.
- Experimental results demonstrate that the proposed power optimization strategy is fast and yields reductions in total power by a factor from 2x to 10x over non-optimization case in nanometer (below 0.1 micron) technologies, maintaining the circuit speed.

Figure 67 shows how all the parts of the dissertation fit into the goal of a complete digital systems design.

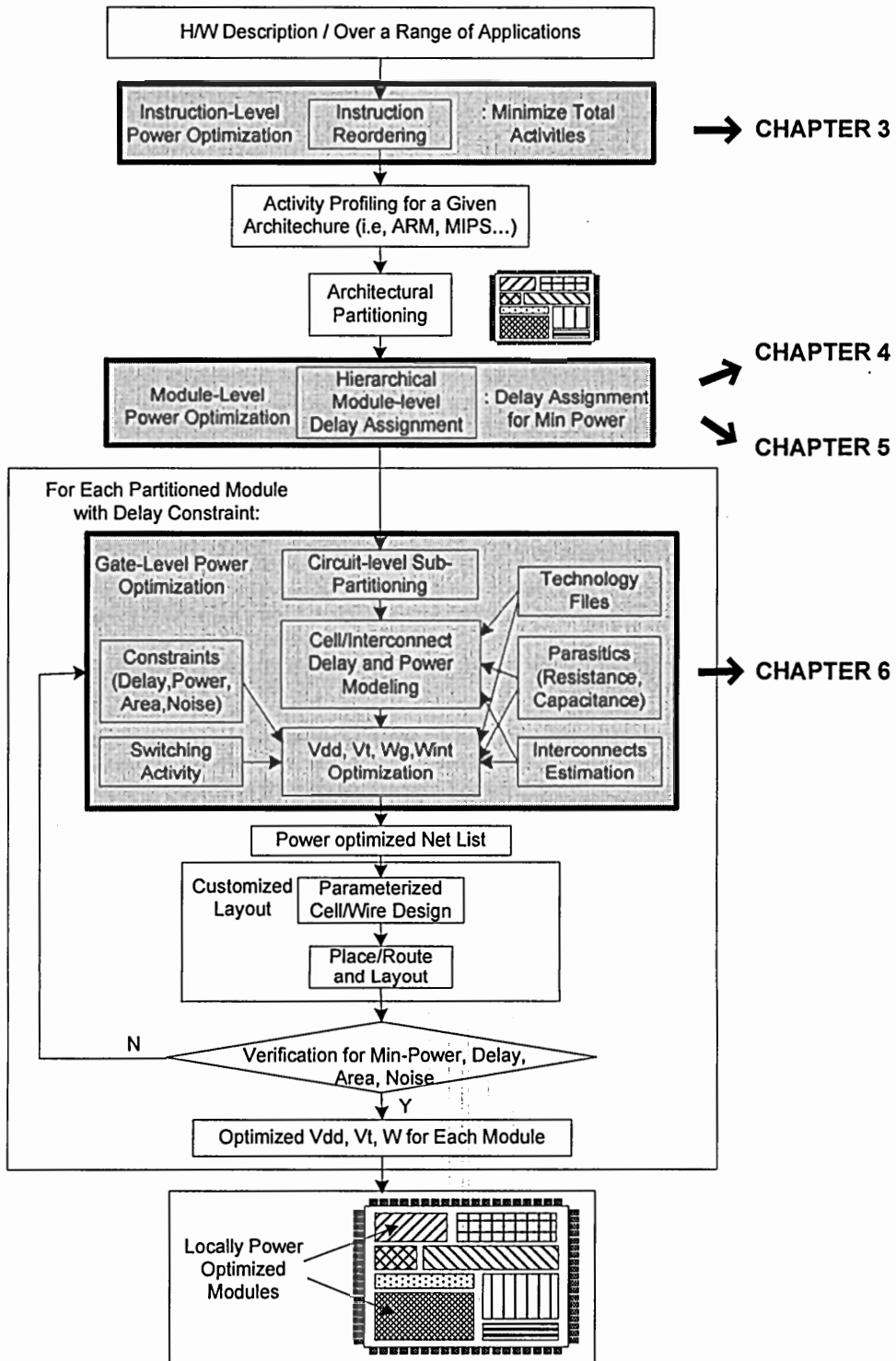


Figure 67. New Design Flow

The research contributions for low-power design during my PhD program have been described in the following publications.

- **K-w. Choi** and A. Chatterjee, "HiPOS: Hierarchical Power Optimization Strategy for ultra low-power CMOS VLSI," submitted to *IEEE Transactions of VLSI Systems*, 2003.
- **K-w. Choi** and A. Chatterjee, "Gate-level power-aware optimization via graph-based timing analysis for ultra low-power CMOS VLSI," submitted to *IEEE/ACM Transactions on Design Automation of Electronic Systems(TODAES)*, 2003.
- **K-w. Choi** and A. Chatterjee, "Efficient Instruction-level optimization methodology for low-power embedded systems," *Proc. of ACM/IEEE International Symposium on System Synthesis*, pp.147-152, Sept. 2001.
- **K-w. Choi**, Y. S. Dhillon, U. Diril, A. Chatterjee, et. Al., "Power-PerformanceTrade-offs in Second Level memory used by as ARM-like RISC Architecture", (one chapter in) *Power Aware Computing*, Kluwer Academic Publishers, Part IV, Chapter 11, pp 215-228, 2001.
- K. Puttasuwami, **K-w. Choi**, A. Chatterjee, et. Al, "System level power-performance trade-offs in embedded systems using voltage and frequency scaling of off-chip buses and memory," *Proc. of ACM/IEEE International Symposium on System Synthesis*, Oct. 2002.
- Y. S. Dhillon, U. Diril, **K-w. Choi**, and A. Chatterjee,"An O(n) supply voltage assignment for low-energy serially connected CMOS modules and a heuristic extension for acyclic data flow graphs", *Proc. of IEEE Computer Society Annual Symposium on VLSI*, 2003.
- **K-w. Choi**, A. Y. S. Dhillon, U. Diril, and A. Chatterjee, "Software-Hardware delay optimization for ultra-low power operations," *GIT-CC-02-16, Technical Report at Georgia Institute of Technology*, pp. 173-179, 2002.
- J.C. Park, **K-w. Choi**, A. Chatterjee, et. Al, "Energy minimization of a pipelined processor using a low voltage pipelined cache," *Proc. of 36th Annual Asilomar Conference on Signals, Systems, and Computers*, Nov. 3-6, 2002.
- **K-w. Choi** and A. Chatterjee, "HA²TSD: Hierarchical time slack distribution for ultra-low power CMOS VLSI" *Proc. of the International Symposium on Low Power Electronics and Design*, pp.207-212, 2002.
- **K-w Choi** and A. Chatterjee, "PA-ZSA (Power Aware Zero Slack Algorithm): A graph based timing analysis for ultra low-power CMOS VLSI," *Proc. of PATMOS'2002*, pp. 178-187, Sep. 2002.
- **K-w. Choi** and A. Chatterjee, "UDSM (ultra deep submicron)-aware post-layout device and interconnect co-optimization for ultra low-power CMOS VLSI," *Proc. of ISLPED*, Aug. 25-27, Seoul, Korea, 2003.

7.2 Future Directions

The open vistas specific to this dissertation are the following:

- Explore the other compiler optimization schemes at the software-level (system-level) to minimize switching activity of the target system.
- Investigate cooperation between a technology level and a circuit level: when the device is scaled and the supply voltage is reduced, it will become very hard to suppress the stand-by leakage power by device itself.
- Consider the systematic and random variations in process, V_{dd} , V_{th} , and temperature are posing a major challenge to the future high-performance and ultra-low-power design. Recently, Intel presents that the spread in frequency and leakage distributions is due to variation in transistor parameters, causing about 20x variation in leakage and 30% variation in chip frequency in a wafer (180nm technology).
- Not only subthreshold leakage but also other leakage components (i.e., gate tunneling or junction leakage,...) should be considered.
- Total leakage (leakages in stand-by mode + leakages in active mode) should be optimized for minimizing total power (switching, short-circuit, and leakage power).
- Judiciously re-consider the traditional approaches according to the device technology scaling (i.e., ultra low-voltage (below 0.5 volt) operation, short-channel effects,...)

BIBLIOGRAPHY

CHAPTER 1

- [1-1] International Technology Roadmap for Semiconductors 2001 Edition, Executive Summary, (<http://public.itrs.net/Files/2001ITRS/Home.htm>).
- [1-2] P.J.M. Havinga and G.J.M. Smit, "Design techniques for low power systems," *Journal of Systems Architecture*, vol. 46, Iss. 1, pp. 745-758, 2000.
- [1-3] J. Meindl, "Low power microelectronics: Retrospect and Prospect," *Proceedings Of the IEEE*, vol. 83, pp. 619-635, April 1995.
- [1-4] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, April 1992.
- [1-5] J.M. Rabaey and M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996.
- [1-6] G. Yeap, *Practical Low Power Digital design*, Kluwer Academic Publishers, 1998.
- [1-7] K. Roy and S. Prasad, *Low-Power CMOS VLSI Circuit Design*, John Wiley & Sons, 2000.
- [1-8] R. Nair, C.L. Berman, P.S. Hauge, and E.J. Yoffe, "Generation of performance constraints for layout," *IEEE Transactions on Computer-Aided Design*, pp.860-874, Aug. 1989.
- [1-9] T. Gao, P.M. Vaidya, and C.L. Liu, "A new performance driven placement algorithm," *Proc. of ICCAD*, pp. 44-47, 1991.
- [1-10] H. Youssef and E. Shragowitz, "Timing constraints for correct performance," *Proc. of ICCAD*, pp. 24-27, 1990.
- [1-11] C. Chen, X. Yang, and M. Sarrafzadeh, "Potential slack: an effective metric of combinational circuit performance," *Proc. of ICCAD*, pp. 198-201, 2000.
- [1-12] T. Kobayashi and T. Sakurai, "Self adjusting threshold voltage scheme (SATS) for low voltage high speed operation," *IEEE CICC*, 1994, pp.271-277.
- [1-13] S. Mutoh, "1-V Power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 847-, April 1992.

- [1-14] A. Fariborz, "A dynamic threshold voltage MOSFET (DTMOS) for ultra-low voltage operation," *IEDM Tech.*, 1994, pp.809-818.
- [1-15] L. Wei, Z. Chen, and K.Roy, "Double gate dynamic threshold voltage (DGDT) SOI MOSFETs for low power high performance designs," *IEEE SOI conference*, 1997, pp. 82-83.
- [1-16] N. Weste and K. Eshraghian, *Principles of CMOS VLSI Design; A Systems Perspective*, Second Ed., Addison-Wesley, pp. 585-588, 1993.
- [1-17] S.S. Sapatnekar, V.B. Rao, P.M. Vaidya, and S Kang, "An exact solution to the transistor sizing problem ofr CMOS circuits using convex optimization," *IEEE Trans. On CAD of Integrated Circuits and Systems*, vol. 12, no. 11, pp. 1621-1634, September 1993.
- [1-18] J. Cong, "An interconnect-centric design flow for nanometer technologies," *Proceedings of the IEEE*, vol. 89, pp. 505-528, April 2001.
- [1-19] T. Sakurai, "Closed-form expression for interconnect delay, coupling, and crosstalk in VLSI," *IEEE Transactions on Electron Devices*, vol. 40, pp. 118-124, Jan. 1993.

CHAPTER 2

- [2-1] International Technology Roadmap for Semiconductors 2001 Edition, Executive Summary, (<http://public.itrs.net/Files/2001ITRS/Home.htm>).
- [2-2] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, April 1992.
- [2-3] J.M. Rabaey and M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996.
- [2-4] G. Yeap, *Practical Low Power Digital design*, Kluwer Academic Publishers, 1998.
- [2-5] A.Raghunathan, N.K. Jha, and S. Dey, *High-Level Power Analysis and Optimization*, Kluwer Academic Publishers, 1998.
- [2-6] K. Roy and S. Prasad, *Low-Power CMOS VLSI Circuit Design*, John Wiley & Sons, 2000.
- [2-7] J. Meindl, "Low power microelectronics: Rtrospect and Prospect," *Proceedings. Of the IEEE*, vol. 83, pp. 619-635, April 1995.
- [2-8] P.J.M. Havinga and G.J.M. Smit, "Design techniques for low power systems," *Journal of Systems Architecture*, vol. 46, Iss. 1, pp. 745-758, 2000.
- [2-9] M. Srivastava, "Designing energy effuicent mobile systems," *Tutorial on MobiCom '99*, Aug., 1999.

- [2-10] T. Sakurai and A.R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE J. Solid-State Circuits*, vol.25, pp. 584-594, Apr. 1990.
- [2-11] F. Najm, "Transition density: A stochastic measure of activity in digital circuits," *Proc. of ACM/IEEE Design Automation Conference. (DAC)*, pp. 644-649, Jun 1991.
- [2-12] M. Pedram, "Power estimation and optimization at the logic level," *International Journal of High Speed Electronics and Systems*, vol. 5, pp. 179-202, Jun 1994.
- [2-13] T. Chou, K. Roy, and S. Prasad, "Estimation of circuit activity considering signal correlation and simultaneous switching," *International Conference on Computer-Aided Design*, pp. 300-303, Nov 1994.
- [2-14] F. Najm, "A survey of power estimation techniques in vlsi circuits," *IEEE Transactions on VLSI Systems*, vol. 2, pp. 446-455, Dec 1994.
- [2-15] H.J.M. Veendrick, "Short-Circuit Dissipation of Static CMOS Circuitry and Its Impact on the Design of Buffer Circuits," *IEEE Journal of Solid-State Circuits*, vol. 19, no. 4, pp. 468-473, Aug. 1984.
- [2-16] E. Macii, M. Pedram, and F. Somenzi, "High-level power modeling, estimation, and optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 17, pp. 1061 -1079, Nov. 1998.
- [2-17] C. Chen, A. Srivastava, M. Sarrafzadeh, "On gate level power optimization using dual-supply voltages," *IEEE Transactions on VLSI systems*, vol. 9, no.5, pp. 616-629, Oct. 2001.
- [2-18] K. Usami and M. Horowitz, "Cluster voltage scaling technique for low power design," *Proc. International Symposium of Low Power Design (ISLPD)*, pp. 3-8, Apr. 1995.
- [2-19] S. Mutoh et al., "1-V power supply high-speed digital circuit technology with multi-threshold voltage CMOS," *IEEE J. Solid-State Circuits*, vol. 30, pp. 847-853, Aug. 1995.
- [2-20] S. Raje and M. Sarrafzadeh, "Scheduling with multiple voltages," *Integration VLSI J.* 23, pp. 37-59, 1997.
- [2-21] J. M. Chang and M. Pedram, "Energy minimization using multiple supply voltages," *IEEE Transactions on VLSI Systems*, vol. 5, pp. 1-8, Dec. 1997.
- [2-22] C. Yeh, Y. Kang, S. Shieh, and J. Wang, "Layout techniques supporting the use of dual supply voltages for cell-based designs," *Proc.of ACM/IEEE Design Automation Conference (DAC)* , pp. 62-67, June 1999.
- [2-23] V. Sundararajan and K. K. Parhi, "Synthesis of low power CMOS VLSI circuits using dual supply voltages," in *Proc. ACM/IEEE Design Automation Conf. (DAC)* , pp. 72-75, June 1999.

- [2-24] K. Usami, M. Igarashi, F. Minami, T. Ishikawa, M. Kanazawa, M. Ichida, K. Nogami, "Automated low-power technique exploiting multiple supply voltages applied to a media processor," *IEEE Journal of Solid-State Circuits*, vol. 33, no. 3, pp. 463-472, Mar. 1999.
- [2-25] D. Marculescu, "Power efficient processors using multiple supply voltages," *Workshop on Compilers and Operating Systems for Low Power (with PACT 2000)*, 2000.
- [2-26] R. Min, T. Furrer, and A. Chandrakasan, "Dynamic Voltage Scaling Techniques for Distributed Microsensor Networks," *Workshop on VLSI (WVLSI '00)*, April 2000.
- [2-27] M. Hashimoto and H. Onodera, "Post-Layout transistor sizing for power reduction in cell-based design," *Proc. of ACM/IEEE Design Automation Conference. (ASP-DAC)*, pp. 359-365, 2001.
- [2-28] K. Kasamsetty, M. Ketkar, A. A. Sapatnekar, "A new class of convex functions for delay modeling and its application to the transistor sizing problem," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, no. 7, pp. 779-788, Jul. 2000.
- [2-29] C. Chen and M. Sarrafzadeh, "Power reduction by simultaneous voltage scaling and gate sizing," *Proc. Asia and South Pacific Design Automation Conf. (ASPDAC)*, pp. 333-338, Jan. 2000.
- [2-30] J. P. Fishburn and A. E. Dunlop, "TILOS: A posynomial programming approach to transistor sizing," *Proc. in International Conference on Computer Aided Design*, pp. 326-328, Nov 1985.
- [2-31] Z. Dai and K. Asada, "MOSIZ: A two-step transistor sizing algorithm based on optimal timing assignments method for multi-stage complex gates," *Proc. of Custom Integrated Circuits Conference*, pp. 17.3.1-17.3.4, May 1989.
- [2-32] S. S. Sapatnekar and V. B. Rao, "iDEAS: A delay estimator and transistor sizing tool for CMOS circuits," *Proc. of Custom Integrated Circuits Conference*, pp. 9.3.1-9.3.4, May 1990.
- [2-33] H. Y. Chen and S. M. Kang, "iCOACH: A circuit optimization aid for CMOS high performance circuits," *Integration, The VLSI Journal*, vol. 10, pp. 185-212, Jan 1991.
- [2-34] L. S. Heusler and W. Fichtner, "Transistor sizing for large combinational digital CMOS circuits," *Integration, The VLSI Journal*, vol.10, pp.155-168, Jan 1991.
- [2-35] S. S. Sapatnekar, V. B. Rao, P. M. Vaidya, and S. M. Kang, "An exact solution to the transistor sizing problem for CMOS circuits using convex optimization," *IEEE Transactions on Computer-Aided Design*, vol. 12, pp. 1621-1632, Nov 1993.

- [2-36] K. Roy, S. Nag, and S. Dutta, "ASAP: A transistor sizing tool for speed, area, and power optimization of static CMOS circuits," Proc. in IEEE International Symposium on Circuits and Systems, pp. 61-64, 1994.
- [2-37] S. S. Sapatnekar and W. Chuang, "Power vs. delay in gate sizing: Conflicting objectives?," Proc. in International Conference on Computer-Aided Design, pp. 463-466, 1995.
- [2-38] P. Pant, V. De, and A. Chatterjee, "Simultaneous power supply, threshold voltage and transistor size optimization for low power operation of CMOS circuits," IEEE Transactions on VLSI Systems, vol. 6, pp. 538-545, Dec 1998.
- [2-39] R. Gogenmoser and H. Kaeslin, "The impact of transistor sizing on power efficiency in submicron CMOS circuits," IEEE Journal of Solid-State Circuits, vol. 32, no. 7, pp. 1142-1145, Jul. 1997.
- [2-40] R. Hossain, M. Zheng, and A. Albicki, "Reducing power dissipation in CMOS circuit by signal probability based transistor reordering," IEEE Transactions on CAD of Integrated Circuits and Systems, vol. 15, no. 3, pp. 361-368, Mar. 1996.
- [2-41] R. Hossain, M. Zheng, and A. Albicki, "Reducing power dissipation in serially connected MOSFET circuits," Proc. of International Conference on Computer Design, pp. 614-617, 1994.
- [2-42] E. Musol and J. Cortadella, "Optimizing CMOS circuits for low power using transistor reordering," Proc. of European Design and Test Conference, pp. 219-223, 1996.
- [2-43] A. Glebov, D. Blaauw, and L. Jones, "Transistor reordering of low power CMOS gates using an SP-BDD representation," Proc. of International Symposium on Lower Power Design, pp. 161-166, 1995.
- [2-44] M. Tachibana, et al., "Power and area optimization by reorganizing CMOS complex gate circuits," Proc. of International Symposium on Lower Power Design, pp. 155-160, 1995.
- [2-45] S. C. Prasad and K. Roy, "Circuit optimization for minimization of power consumption under delay constraints," Proc. of International Workshop on Low Power Design, pp. 15-20, Apr. 1994.
- [2-46] J. Kao and A. Chandrakasan, "Dual-threshold voltage techniques for low-power digital circuits," IEEE Journal of Solid-State Circuits, vol. 35, iss. 7, pp.1009-1018, Jul. 2000.
- [2-47] J. Kao, A. Chandrakasan, and D. Antoniadis, "Transistor sizing issues and tool for multi-threshold CMOS Technology," Proc. ACM/IEEE Design Automation Conf. (DAC), pp. 409-414, 1999.
- [2-48] S. Sirichotiyakul, et al., "Stand-by power minimization through simultaneous threshold voltage selection and circuit sizing," Proc. ACM/IEEE Design Automation Conf. (DAC) , pp. 436-441, 1999.

- [2-49] Q. Wang and S. B. K. Vrudhula, "An investigation of power delay trade-offs for dual V_t CMOS circuits," Proc. International Conference on Computer Design, pp. 556-562, 1999.
- [2-50] L. Wei, Z. Chen, K. Roy, Y. Ye, and V. De, "Mixed- V_{th} (MVT) CMOS circuit design methodology for low power applications," Proc. ACM/IEEE Design Automation Conf. (DAC), pp. 430-435, 1999.
- [2-51] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada, "1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS," IEEE Journal of Solid-State Circuits, vol. 30-8, pp. 847-854, Aug. 1995.
- [2-52] T. Kobayashi and T. Sakurai, "Self adjusting threshold voltage scheme (SATS) for low voltage high speed operation," Proc. of Custom Integrated Circuits Conference, pp.271-274, 1994.
- [2-53] A. Fariborz, "A dynamic threshold voltage MOSFET (DTMOS) for ultra-low voltage operation," IEDM Tech., pp. 809-816, 1994.
- [2-54] L. Wei, Z. Chen, and K. Roy, "Double gate dynamic threshold voltage (DGDT) SOI MOSFETs for low power high performance designs," Proc. of IEEE SOI conference, pp. 82-83 1997.
- [2-55] D. Liu and C. Svensson, "Trading speed for low power by choice of supply and threshold voltages," IEEE Journal of Solid-State Circuits, vol. 28, pp. 10-17, Jan 1993.
- [2-56] Z. Chen and J. Plummer, "Low threshold voltage quarter micron MOSFETs for low power applications," Proc. of IEEE Symposium on Low Power Electronics, pp.78-79, 1995.
- [2-57] R. Gonzalez, B. M. Gordon, and M. Horowitz, "Supply and threshold voltage scaling for low power CMOS," IEEE Journal of Solid-State Circuits, vol.32, pp.1210-1216, Aug 1997.
- [2-58] Z. Chen, C. Diaz, J. Plummer, M. Cao, and W. Greene, "0.18 μ m dual-vt MOSFET process and energy-delay measurement," International Electron Devices Meeting, pp.851-858, 1996.
- [2-59] L. Wei, Z. Chen, M. Johnson, K. Roy, and V. De, "Design and optimization of low voltage high performance dual threshold CMOS circuits," Proc. ACM/IEEE Design Automation Conf. (DAC), pp. 489-494, 1998.
- [2-60] Q. Wang and S. Vrudhula, "Static power optimization of deep submicron CMOS circuits for dual vt technology," Proc. of International Conference on Computer-Aided Design, pp.490-494, 1998.
- [2-61] K. S. Khouri and N. K. Jha, "Leakage power analysis and reduction during behavioral synthesis," Proc. of International Conference of Computer Design, pp. 561-564, 2000.

- [2-62] G. D. Micheli, *Synthesis and Optimization of Digital Circuits*, McGraw Hill, 4 ed., 1994.
- [2-63] H. Savoj, R. Brayton, and H. Touati, "Extracting Local Don't-Cares for Network Optimization," *Proc. of the International Conference on Computer-Aided Design*, pp. 514–517, Nov. 1991.
- [2-64] A. Shen, S. Devadas, A. Ghosh, and K. Keutzer, "On Average Power Dissipation and Random Pattern Testability of Combinational Logic Circuits," *Proc. of the International Conference on Computer-Aided Design*, pp. 402–407, Nov. 1992.
- [2-65] S. Iman and M. Pedram, "Multi-Level Network Optimization for LowPower," *Proc. of the International Conference on Computer-Aided Design*, pp. 371–377, Nov. 1994.
- [2-66] C. Lemonds and S. S. Mahant-Shetti, "A Low Power 16 by 16 Multiplier Using Transition Reduction Circuitry," *Proc. of the International Workshop on Low Power Design*, pp. 139–142, Apr. 1994.
- [2-67] P. Pant, V. De, and A. Chatterjee, "Device-circuit optimization for minimal energy and power consumption in CMOS random logic networks," *Proc. ACM/IEEE Design Automation Conf. (DAC)*, pp. 403-408, Jun. 1997.
- [2-68] K. Roy and S. C. Prasad, "Circuit activity based logic synthesis for low reliable operation," *IEEE Transactions on VLSI Systems*, vol. 1, pp. 503-513, Dec. 1993.
- [2-69] R. K. Brayton and C. McMullen, "The decomposition and factorization of Boolean expressions," *Proc. International symposium on Circuit and Systems*, pp. 49-54, May 1992.
- [2-70] C. Yeh, C. C. Chang, and J. Wang, "Technology mapping for low power," *Proc. of ACM/IEEE Design Automation Conf. (ASP-DAC)*, pp. 145-148, 1999.
- [2-71] R. Murgai, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Decomposition of logic functions for minimum transition activity," *Proc. of the European Design and Test Conference*, pp. 404-410, 1995.
- [2-72] V. Tiwari, P. Ashar, and S. Malik, "Technology mapping for low power," *Proc. of ACM/IEEE Design Automation Conf. (DAC)*, pp. 74-79, Jun. 1993.
- [2-73] K. Keutzer, "Technology mapping and local optimization," *Proc. of ACM/IEEE Design Automation Conf. (DAC)*, pp. 341-347, Jun. 1987.
- [2-74] C-Y Tsui, M. Pedram, and A. M. Despain, "Technology decomposition and mapping targeting low power dissipation," *Proc. of ACM/IEEE Design Automation Conf. (DAC)*, pp. 68-73, Jun. 1993.
- [2-75] C-Y. Tsui, M. Pedram, C-A. Chen, and A. M. Despain, "Low Power State Assignment Targeting Two- and Multi-level Logic Implementations," *Proc. of the International Conference on Computer-Aided Design*, pp. 82-87, Nov. 1994.

- [2-76] G. D. Hachtel, M. Hermida, A. Pardo, M. Poncino, and F. Somenzi, "Re-Encoding Sequential Circuits to Reduce Power Dissipation," Proc. of the International Conference on Computer-Aided Design, pp. 70-73, Nov. 1994.
- [2-77] M. Stan and W. Burleson, "Limited-weight codes for low-power I/O," Proc. of the International Workshop on Low Power Design, pp. 209-214, Apr. 1994.
- [2-78] M. Stan and W. Burleson, "Bus invert coding for low-power I/O," IEEE Transactions on VLSI Systems, vol. 3, no. 1, pp. 49-58, Mar. 1995.
- [2-79] J. Monteiro, S. Davadas, and A. Ghosh, "Retiming sequential circuits for low power," Proc. of the IEEE International Conference on Computer Aided Design, pp. 398-402, Nov. 1993.
- [2-80] C. E. Leiserson, F. M. Rose, and J. B. Saxe, "Optimizing Synchronous Circuitry by Retiming," Proc. of 3rd CalTech Conference on VLSI, pp 23-36, Mar. 1983.
- [2-81] R. K. Ranjan, V. Singhal, F. Somenzi, and R. K. Brayton, "On the optimization power of retiming and resynthesis transformations," Proc. of International Conference on Computer-Aided Design, PP 402-407, 1998.
- [2-82] Y. Xia and A. E. A. Almaini, "Differential CMOS edge-triggered flip-flop with clock-gating," IEEE Electronic Letters, vol. 38, 3rd, pp. 9-11, Jan. 2002.
- [2-83] Q. Wu, M. Pedram, and X. Wu, "Clock-gating and its application to low power design of sequential circuits," IEEE Transactions on Circuits and Systems, vol. 47, iss. 3, pp.415-420,Mar. 2000.
- [2-84] A. G. M. Strollo, E. Napoli, and D. DeCaro, "Now clock-gating techniques for low-power flip-flops," ISLPED '00, pp. 114-119, 2000.
- [2-85] T. Kitahara, F. Minami, T. Ueda, K. Usami, S. Nishio, M. Murakata, and T. Mitsuhashi, "A clock-gating method for low-power LSI design," Proc. of the ASP-DAC '98, pp. 307-312, 1998.
- [2-86] M. Alidina, J. Monteiro, S. Devadas, A. Ghosh, and M. Papaefthymiou, "Precomputation-Based Sequential Logic Optimization for Low Power," IEEE Transactions on VLSI Systems,vol. 2, no. 4, pp. 426-436, Dec. 1994.
- [2-87] J. Monteiro, J. Rinderknecht, S. Devadas, and A. Ghosh, "Optimization of Combinational and Sequential Logic Circuits for Low Power Using Precomputation," Proc. of the Chapel Hill Conference on Advanced Research on VLSI, Mar. 1995.
- [2-88] V. Tiwari, S. Malik, and P. Ashar, "Guarded Evaluation: Pushing Power Management to Logic Synthesis/Design," Proc. of the International Symposium on Low Power Design, Apr. 1995.
- [2-89] T. Ishihara and K. Asada, "A system level memory power optimization technique using multiple supply and threshold voltages," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 456-461, 2001.

- [2-90] L. Benini, A. Macii, E. Macii, and M. Poncino, "Synthesis of application-specific memories for power optimization in embedded systems," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 456-461, 2000.
- [2-91] C. Su, A. Despain, "Cache Design Tradeoffs for Power and Performance Optimization: a Case Study," Proc. of International Symposium on Low Power Design, pp. 63-68, April 1995.
- [2-92] R. Panwar, D. Rennels, "Reducing the Frequency of Tag Compares for Low Power I-Cache Design," Proc. of International Symposium on Low Power Design, pp. 57-62, April 1995.
- [2-93] M. Kamble, K. Ghose, "Analytical Energy Dissipation Models for Low Power Caches," Proc. of International Symposium on Low Power Electronics and Design, pp. 143-148, Aug. 1997.
- [2-94] T. Juan, T. Lang, J. Navarro, "Reducing TLB Power Requirements," Proc. of International Symposium on Low Power Electronics and Design, pp. 196-201, Aug. 1997.
- [2-95] R. Bajwa, M. Hiraki, H. Kojima, et al., "Instruction Buffering to Reduce Power in Processors for Signal Processing," IEEE Transactions on VLSI Systems, vol. 5, no. 4, pp. 417-424, Dec. 1997.
- [2-96] J. Kin, M. Gupta and W. Mangione-Smith, "The Filter Cache: An Energy-Efficient Memory Structure," Proc. of IEEE International Symposium on Microarchitecture, pp. 184-193, Dec. 1997.
- [2-97] C. Kulkarni, F. Catthoor, H. De Man, "Code Transformations for Low Power Caching in Embedded Multimedia Processors," Proc. of International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing, pp. 23-26, March 1998.
- [2-98] U. Ko and P. Balsara, "Energy Optimization of Multilevel Cache Architectures for RISC and CISC Processors," IEEE Transactions on VLSI Systems, vol. 6, no. 2, pp. 299-308, June 1998.
- [2-99] R. Bahar, G. Albera, S. Manne, "Power and Performance Tradeoffs using Various Caching Strategies," Proc. of International Symposium on Low Power Electronics and Design, pp. 64-69, Aug. 1998.
- [2-100] N. Bellas, I. Hajj, C. Polychronopoulos, G. Stamoulis, "Architectural and Compiler Support for Energy Reduction in the Memory Hierarchy of High Performance Microprocessors," Proc. of International Symposium on Low Power Electronics and Design, pp. 64-69, Aug. 1998.
- [2-101] D. Lidsky, J. Rabaey, "Low-Power Design of Memory Intensive functions," Proc. of Symposium on Low Power Electronics, pp. 16-17, Oct. 1994.

- [2-102] M. Lee, V. Tiwari, "A Memory Allocation Technique for Low-Energy Embedded DSP Software," Proc. of Symposium on Low Power Electronics, pp. 24-25, 1995.
- [2-103] R. Saied, C. Chakrabarti, "Scheduling for Minimizing the Number of Memory Accesses in Low Power Applications," VLSI Signal Processing, pp. 169-178, Oct. 1996.
- [2-104] C. Gebotys, "Low Energy Memory and Register Allocation Using Network Flow," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp 435-440, June 1997.
- [2-105] S. Wuytack, F. Catthoor, H. De Man, "Transforming Set Data Types to Power Optimal Data Structures," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 15, no. 6, pp. 619-629, June 1996.
- [2-106] L. Nachtergaele, D. Moolenaar, B. Vanhoof, F. Catthoor, H. De Man, "System-Level Power Optimization of Video Codecs on Embedded Cores: a Systematic Approach," Journal of VLSI Signal Processing, vol. 18, no. 2, pp. 89-109, Feb. 1998.
- [2-107] E. De Greef, F. Catthoor, H. De Man, "Program Transformation Strategies for Memory Size and Power Reduction of Pseudo regular Multimedia Subsystems," IEEE Transactions on Circuits and Systems for Video Technology, vol. 8, no. 6, pp. 719-733, Oct. 1998.
- [2-108] S. Wuytack, J. Diguët, F. Catthoor, H. De Man, "Formalized Methodology for Data Reuse: Exploration for Low-Power Memory Mappings," IEEE Transactions on VLSI Systems, vol. 6, no. 4, pp. 529-537, Dec. 1998.
- [2-109] V. Zyuban, P. Kogge, "The Energy Complexity of Register Files," Proc. of International Symposium on Low Power Electronics and Design, pp. 305-310, Aug. 1998.
- [2-110] D. Kirovski, C. Lee, M. Potkonjak, W. Mangione Smith, "Synthesis of Power Efficient Systems-on-Silicon," Proc. of Asian and South Pacific Design Automation Conference, pp. 557-562, Feb. 1998.
- [2-111] Y. Li and J. Henkel, "A Framework for Estimating and Minimizing Energy Dissipation of Embedded HW/SW Systems," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp.188-193, June 1998.
- [2-112] S. Coumeri, D. Thomas, "Memory Modeling for System Synthesis," Proc. of International Symposium on Low Power Electronics and Design, pp. 179-184, June 1998.
- [2-113] J. Da Silva, F. Catthoor, F. Verkest, H. De Man, "Power Exploration for Dynamic Data Types through Virtual Memory Management Refinement," Proc. of International Symposium on Low Power Electronics and Design, pp. 311-316, Aug. 1998.

- [2-114] T. Simunic, L. Benini, G. De Micheli, "Cycle-Accurate Simulation of Energy Consumption in Embedded Systems," Proc. of ACM/IEEE Design Automation Conference. (DAC), June 1999.
- [2-115] G. Esakkimuthu, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin, "Memory system energy: Influence of hardware-software optimizations," Proc. of International Symposium on Low Power Electronics and Design, pp. 244-246, 2000.
- [2-116] J. Brown, D. Chen, G. Greenwood, X. Hu, R. Taylor, "Scheduling for Power Reduction in a Real-Time System," Proc. of International Symposium on Low Power Electronics and Design, pp. 84-87, Aug. 1997.
- [2-117] D. Kirowski, M. Potkonjak, "System-level synthesis of low-power Hard Real-Time Systems," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 697-702, June 1997.
- [2-118] B. Dave, G. Lakshminarayana, N. Jha, "COSYN: Hardware-Software Co-Synthesis for Heterogeneous Distributed Embedded Systems," IEEE Transactions on VLSI Systems, vol. 7, no. 1, March 1999.
- [2-119] M. Wan, Y. Ichikawa, D. Lidsky, J. Rabaey, "An Energy Conscious Methodology for Early Design Exploration of Heterogeneous DSPs," Proc. of Custom Integrated Circuits Conference, pp. 111-117, May 1998.
- [2-120] J. Henkel, "A Low-Power Hardware/Software Partitioning Approach for Core-based Embedded Systems," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 122-127, June 1999.
- [2-121] J. L. Hennessy and D. A. Patterson, Computer Architecture: A Quantitative Approach, Second Ed., Morgan Kaufmann Publishers, 1996.
- [2-122] A. Chandrakasan, et al., "A low power chipset for portable multimedia applications," Proc. of ISSCC, pp. 82-83, Feb 1994.
- [2-123] N. Nishi, et al., "A 1GIPS 1W single-chip tightly-coupled four-way multiprocessor with architecture support for multiple control flow execution," Proc. of ISSCC 2000 Digest of Technical papers, TP14.1, Feb. 2000.
- [2-124] V. Tiwari, S. Malik, A. Wolfe, "Power Analysis of Embedded Software: A First Step Towards Software PowerMinimization," IEEE Transactions on VLSI, vol. 2, no.4, pp.437-445, Dec. 1994.
- [2-125] V. Tiwari, S. Malik, A. Wolfe, M. Lee, "Instruction Level Power Analysis and Optimization of Software," Journal of VLSI Signal Processing, vol. 13, no.1-2, pp.223-233, 1996.
- [2-126] H. Mehta, R. Owens, M. Irwin, R. Chen, D. Ghosh, "Techniques for Low Energy Software," Proc. of International Symposium on Low Power Electronics and Design, pp. 72-75, Aug 1997.

- [2-127] A. Kalambur, M. Irwin, "An Extended Addressing Mode for Low Power," Proc. of International Symposium on Low Power Electronics and Design, pp. 208-213, Aug. 1997.
- [2-128] C. Gebotys, R. Gebotys, "An Empirical Comparison of Algorithmic, Instruction, and Architectural Power Prediction Model for High-Performance Embedded DSP Processors," Proc. of International Symposium on Low Power Electronics and Design, pp. 121-123, Aug. 1998.
- [2-129] A. Hasegawa, I. Kawasaki, K. Yamada, S. Yoshioka, S. Kawasaki, P. Biswas, "SH3: High Code Density, Low Power," IEEE Micro, vol. 15, no. 6, pp. 11-19, Dec. 1995.
- [2-130] S. Segars, K. Clarke, L. Goudge, "Embedded Control Problems, Thumb and the ARM7TDM1," IEEE Micro, vol. 15, no. 5, pp. 22-30, Oct. 1995.
- [2-131] J. Russel, M. Jacone, "Software Power Estimation and Optimization for High-Performance, 32-bit Embedded Processors," Proc. of International Conference on Computer Design, pp. 328-333, Oct. 1998.
- [2-132] Kyu-won Choi and Abhijit Chatterjee, "Efficient Instruction-level optimization methodology for low-power embedded systems," Proc. of ACM/IEEE International Symposium on System Synthesis, pp.47-152, Sept. 2001.
- [2-133] M. Potkonjak, M. Rabaey, "Power Minimization in DSP Application Specific Systems Using Algorithm Selection," Proc. of International Conference on Acoustics, Speech, and Signal Processing, vol. 4, pp. 2639-2642, May 1995.
- [2-134] A. Chandrakasan, M. Potkonjak, R. Mehra, J. Rabaey, R. Brodersen, "Optimizing Power Using Transformations," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 14, no. 1, pp. 12-31, Jan. 1995.
- [2-135] R. Mehra, L. Guerra, J. Rabaey, "Low-Power Architectural Synthesis and the Impact of Exploiting Locality," Journal of VLSI Signal Processing, vol. 13, no. 2-3, pp. 239-58, Aug-Sept. 1996.
- [2-136] M. Srivastava, M. Potkonjak, "Power Optimization in Programmable Processors and ASIC Implementation of Linear Systems: Transformation-Based Approach," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 343-348, June 1996.
- [2-137] R. Mehra, L. Guerra, J. Rabaey, "A Partitioning Scheme for Optimizing Interconnect Power," IEEE Journal of Solid-State Circuits, vol. 32, no. 3, pp. 433-43, March 1997.
- [2-138] D. Kim and K. Choi, "Power-Conscious High-Level Synthesis using Loop Folding," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 441-445, June 1997.

- [2-139] I. Hong, M. Potkonjak, R. Karri, "Power Optimization Using Divide-and-Conquer Techniques for Minimization of the Number of Operations," Proc. of International Conference on Computer-Aided Design, pp. 108-113, Nov. 1997.
- [2-140] G. Lakshminarayana, N. Jha, "FACT: a Framework for the Application of Throughput and Power Optimizing Transformations to Control-Flow Intensive Behavioral Descriptions," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 102-107, June 1998.
- [2-141] L. Guerra, M. Potkonjak, J. Rabaey, "A methodology for guided behavioral-level optimization," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 309-314, June 1998.
- [2-142] T. Simunic, L. Benini, A. Acquaviva, P. Glynn, and G. D. Micheli, "Dynamic voltage scaling and power management for portable systems," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 524-529, June 2001.
- [2-143] V. Tiwari, R. Donnelly, S. Malik, and R. Gonzalez, "Dynamic power manegagment for microprocessors: acase study," Proc. of International Conference on VLSI Design, pp. 185-192, 1997.
- [2-144] A. Farrahi, G. Tellez, M. Sarrafzadeh, "Memory Segmentation to Exploit Sleep Mode Operation," Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 36-41, June 1995.
- [2-145] A. Farrahi, M. Sarrafzadeh, "System Partitioning to Maximize Sleep Time," Proc. of Interantional Conference on Computer-Aided Design, pp. 452-455, Nov. 1995.
- [2-146] G. Tellez, A. Farrahi and M. Sarrafzadeh, "Activity Driven Clock Design for Low-Power Circuits," Proc. of International Conference on Computer-Aided Design, pp. 62-65, Nov. 1995.
- [2-147] P. Krishnan, P. Long and J. Vitter, "Adaptive Disk Spindown Via Optimal Rent-to-buy in Probabilistic Environments," Proc. of International Conference on Machine Learning, pp. 322-330, July 1995.
- [2-148] D. Helmbold, D. Long, E. Sherrod, "Dynamic Disk Spin-down Technique for Mobile Computing," Proc. of Conference on Mobile Computing, pp. 130-142, Nov. 1996.
- [2-149] F. Douglis, P. Krishnan and B. Bershad, "Adaptive Disk Spin-down Policies for Mobile Computers," Proc. of USENIX Symposium on Mobile and Location Independent Computing, pp. 121-137, Apr. 1995.
- [2-150] M. Srivastava, A. Chandrakasan. R. Brodersen, "Predictive System Shutdown and Other Architectural Techniques for Energy Efficient Programmable Computation," IEEE Transactions on VLSI Systems, vol. 4, no. 1, pp. 42-55, March 1996.

- [2-151] C.-H. Hwang and A. Wu, "A Predictive System Shutdown Method for Energy Saving of Event-Driven Computation", Proc. of International Conference on Computer Aided Design, pp. 28-32, Nov. 1997.
- [2-152] E. Chung, L. Benini, A. Bogliolo and G. De Micheli, "Dynamic Power Management for Non-Stationary Service Requests," Design and Test in Europe Conference, March 1999, pp. 77-81.
- [2-153] L. Benini, A. Bogliolo, G. Paleologo and G. De Micheli, "Policy Optimization for Dynamic Power Management," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 18, no. 6, pp. 813-833, June 1999.
- [2-154] Q. Quiu and M. Pedram, "Dynamic Power Management Based on Continuous-Time Markov Decision Processes." Proc. of ACM/IEEE Design Automation Conference. (DAC), pp. 555-561, June 1999.
- [2-155] E. Vittoz, "Low Power Microelectronics: Ways to Approach the Limits," Proc. of International Solid-State Circuits Conference, pp. 14-18, Jan. 1994.
- [2-156] S. Nawab, A. Oppenheim, A. Chandrakasan, J. Winograd, J. Ludwig, "Approximate Signal Processing," Journal of VLSI Signal Processing, vol. 15, no. 1-2, pp. 177-200, Jan. 1997.
- [2-157] R. Hedge, N. Shanbhag, "Energy-Efficiency in Presence of Deep Submicron Noise," Proc. of International Conference on Computer-Aided Design, pp. 228-234, Nov. 1998.
- [2-158] C.-L. Su, C.-Y. Tsui, and A. M. Despain, "Low power architecture design and compilation techniques for high-performance processors," *Proc. IEEE CompCon'94*, pp. 489-498. Feb., 1994.
- [2-159] H. Tomiyama, T. Ishihara, A. inoue, and H. Yasuura, "Instruction Scheduling to Reduce Switching Activity of Off-Chip Busses for Low-Power Systems with Caches," *Proc. IEICE*, pp. 2621-2629. Dec., 1998.
- [2-160] R. Nair, C.L. Berman, P.S. hauge, and E.J. Yoffe, "Generation of performance constraints for layout," *IEEE Transactions on Computer-Aided Design*, pp.860-874, Aug. 1989.
- [2-161] T. Gao, P.M. Vaidya, and C.L. Liu,"A new performance driven placement algorithm," *Proc. of ICCAD*, pp. 44-47, 1991.
- [2-162] C. Chen, X. Yang, and M. Sarrafzadeh, "Potential slack: an effective metric of combinational circuit performance," *Proc. of ICCAD*, pp. 198-201, 2000.
- [2-163] J. Cong, "An interconnect-centric design flow for nanometer technologies," *Proceedings of the IEEE*, vol. 89, pp. 505-528, April 2001.

- [2-164] T. Sakurai, "Closed-form expression for interconnect delay, coupling, and crosstalk in VLSI," *IEEE Transactions on Electron Devices*, vol. 40, pp. 118-124, Jan. 1993.
- [2-165] D. Sylvester and K. Keutzer, "Getting to the bottom of deep submicron," *Proc. of ICCAD*, pp. 203-211, 1998.

CHAPTER 3

- [3-1] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, and A. Schrijver, *Combinatorial Optimization*, Wiley-Interscience Series in Discrete Mathematics and Optimization, pp.241-271, 1998.
- [3-2] V. Jain, S. Rele, S. Pande, and J. Ramanujam "Code restructuring for improving execution efficiency, code size and power consumption for embedded DSPs", *12th International Workshop on Languages and Compilers for Parallel Computing*, 1999.
- [3-3] C.-L. Su, C.-Y. Tsui, and A. M. Despain, "Low power architecture design and compilation techniques for high-performance processors," *Proc. IEEE CompCon '94*, pp. 489-498. Feb., 1994.
- [3-4] V. Tiwari, S. Malik, and A. Wolfe, "Instruction Level Power Analysis and Optimization of Software," *Journal of VLSI Signal Processing*, pp.1-18, 1996.
- [3-5] V. Tiwari, S. Malik, and A. Wolfe, "Compilation Techniques for Low Energy: An Overview," *IEEE Symposium on Low Power Electronics*, October 1994.
- [3-6] H. Tomiyama, T. Ishihara, A. inoue, and H. Yasuura, "Instruction Scheduling to Reduce Switching Activity of Off-Chip Busses for Low-Power Systems with Caches," *Proc. IEICE*, pp. 2621-2629. Dec., 1998.
- [3-7] C. Lee, J.K. Lee, and T.T. Hwang, "Compiler optimization on Instruction Scheduling for Low Power," *ISSS2000*, pp.20-22, Oct., 2000.
- [3-8] K.Roy and M.C. Johnson, "Software design for low power," *Technical report at Purdue Univ.*, 1996.
- [3-9] C.-L. Su, C.-Y. Tsui, and A. M. Despain, "Low power architecture design and compilation techniques for high-performance processors," *Proc. IEEE CompCon '94*, pp. 489-498. Feb., 1994.
- [3-10] G. Lakshminarayana, A. Raghunathan, N.K. Jha, "Incorporating Speculative Execution into Scheduling of Control-Flow-Intensive Designs," *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, vol., 19, no. 3, pp. 308-324 Mar., 2000.
- [3-11] D. Burger and T.M. Austin, "SimpleScalar Tool Set," *Univ. of Wisconsin-Madison Computer Science Dept., Tech. Report #1342*, Jun., 1997.

- [3-12] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D. Shmoys, *The Traveling Salesman Problem*, Wiley, Chichester, 1985.
- [3-13] R.C. Prim, "Shortest connection networks and some generalizations," *Bell System Technical Journal* 36, pp1389-1401, 1957.
- [3-14] K.C. Kapur and L.R. Lamberson, *Reliability in Engineering Design*, John Wiley & Sons, 1977.
- [3-15] T.D. Burd and R.W. Brodersen, "Energy efficient CMOS microprocessor design," *Proceedings 28th HICSS Conference*, vol. I, pp. 288-297, Jan., 1995.

CHAPTER 4

- [4-1] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 473-484, April 1992.
- [4-2] J.M. Rabaey and M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996, pp 21-64,130-160.
- [4-3] R. Nair, C.L. Berman, P.S. hauge, and E.J. Yoffe, "Generation of performance constraints for layout," *IEEE Transactions on Computer-Aided Design*, pp.860-874, Aug. 1989.
- [4-4] T. Gao, P.M. Vaidya, and C.L. Liu,"A new performance driven placement algorithm," *Proc. of ICCAD*, pp. 44-47, 1991.
- [4-5] H. Youssef and E. Shragowitz, "Timing constraints for correct performance," *Proc. of ICCAD*, pp. 24-27, 1990.
- [4-6] C. Chen, X. Yang, and M. Sarrafzadeh, "Potential slack: an effective metric of combinational circuit performance," *Proc. of ICCAD*, pp. 198-201, 2000.
- [4-7] P. Pant, V. De, and A. Chatterjee, "Simultaneous power Supply, threshold voltage, and transistor size optimization for low-power operation of CMOS circuits," *IEEE Trans. On VLSI Systems*, vol. 6, no. 4, pp. 538-545, December 1998.
- [4-8] L. Wei, Z. Chen, K. Roy, M. Johnson, Y. Ye, and V. De, "Design and optimization of dual threshold-circuits for low-voltage low-power applications," *IEEE Trans. On VLSI Systems*, vol. 7, no. 1, pp 16-24, March 1999.
- [4-9] T. Sakurai and A.R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal Solid-State Circuits*, vol. 25, pp. 584-594, Apr. 1990.
- [4-10] 10. N. Hedenstierna and L. Jeppson, "CMOS circuit speed and buffer optimization," *IEEE Trans. On Computer-Aided Design*, vol. 6, pp. 270-281, March, 1987.

- [4-11] 11. A. Bhavnagarwala, V. De, B. Austin, and J. Meindl, "Circuit techniques for CMOS low power GSI," in Proc. Int. Symp. Low Power Electron. Design: Dig. Tech. Papers, Aug. 1996, pp. 193-196.
- [4-12] 12. A.Raghunathan, N.K. Jha, and S. Dey, *High-Level Power Analysis and Optimization*, Kluwer Academic Publishers, 1998, pp 1-25.
- [4-13] 13. K. Roy and S.C. Prasad, *Low-Power CMOS VLSI Circuit Design*, John Wiley & Sons, Inc., 2000, pp 201-252.

CHAPTER 5

- [5-1] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 473-484, April 1992.
- [5-2] J.M. Rabaey and M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996, pp 21-64,130-160.
- [5-3] R. Nair, C.L. Berman, P.S. Hauge, and E.J. Yoffe, "Generation of performance constraints for layout," *IEEE Transactions on Computer-Aided Design*, pp.860-874, Aug. 1989.
- [5-4] T. Gao, P.M. Vaidya, and C.L. Liu, "A new performance driven placement algorithm," *Proc. of ICCAD*, pp. 44-47, 1991.
- [5-5] H. Youssef and E. Shragowitz, "Timing constraints for correct performance," *Proc. of ICCAD*, pp. 24-27, 1990.
- [5-6] P. Pant, V. De, and A. Chatterjee, "Simultaneous power supply, threshold voltage, and transistor size optimization for low-power operation of CMOS circuits," *IEEE Trans. On VLSI Systems*, vol. 6, no. 4, pp. 538-545, December 1998.
- [5-7] T. Sakurai and A.R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal Solid-State Circuits*, vol. 25, pp. 584-594, Apr. 1990.
- [5-8] A. Bhavnagarwala, V. De, B. Austin, and J. Meindl, "Circuit techniques for CMOS low power GSI," in Proc. Int. Symp. Low Power Electron. Design: Dig. Tech. Papers, Aug. 1996, pp. 193-196.
- [5-9] A.Raghunathan, N.K. Jha, and S. Dey, *High-Level Power Analysis and Optimization*, Kluwer Academic Publishers, 1998, pp 1-25.
- [5-10] K. Roy and S.C. Prasad, *Low-Power CMOS VLSI Circuit Design*, John Wiley & Sons, Inc., 2000, pp. 201-252.
- [5-11] T. Kobayashi and T. Sakurai, "Self adjusting threshold voltage scheme (SATS) for low voltage high speed operation," *IEEE CICC*, 1994, pp.271-277.

- [5-12] S. Mutoh, "1-V Power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 847-, April 1992.
- [5-13] A. Fariborz, "A dynamic threshold voltage MOSFET (DTMOS) for ultra-low voltage operation," *IEDM Tech.*, 1994, pp.809-818.
- [5-14] L. Wei, Z. Chen, and K.Roy, "Double gate dynamic threshold voltage (DGDT) SOI MOSFETs for low power high performance designs," *IEEE SOI conference*, 1997, pp. 82-83.
- [5-15] S.S. Sapatnekar, V.B. Rao, P.M. Vaidya, and S Kang, "An exact solution to the transistor sizing problem of CMOS circuits using convex optimization," *IEEE Trans. On CAD of Integrated Circuits and Systems*, vol. 12, no. 11, pp. 1621-1634, September 1993.

CHAPTER 6

- [6-1] International Technology Roadmap for Semiconductors 2001 Edition, Executive Summary, (<http://public.itrs.net/Files/2001ITRS/Home.htm>).
- [6-2] J. Meindl, "Low power microelectronics: retrospect and prospect," *Proceedings of the IEEE*, vol. 83, pp. 619-635, April 1995.
- [6-3] J. Cong, "An interconnect-centric design flow for nanometer technologies," *Proceedings of the IEEE*, vol. 89, pp. 505-528, April 2001.
- [6-4] A. Chandrakasan, S. Sheng, and R. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, pp. 473-484, April 1992.
- [6-5] J.M. Rabaey and M. Pedram, *Low Power Design Methodologies*, Kluwer Academic Publishers, 1996, pp 21-64,130-160.
- [6-6] Pakaj Pant, Vivek De, and Abhijit Chatterjee, "Simultaneous power Supply, threshold voltage, and transistor size optimization for low-power operation of CMOS circuits," *IEEE Transactions on VLSI Systems*, vol. 6, pp. 538-545, Dec 1998.
- [6-7] Kyu-won Choi and Abhijit Chatterjee, "HA²TSD: Hierarchical time slack distribution for ultra-low power CMOS VLSI," *Proc. of the International Symposium on Low Power Electronics and Design*, pp.207-212, 2002.
- [6-8] Kyu-won Choi and Abhijit Chatterjee, "PA-ZSA (Power Aware Zero Slack Algorithm): A graph based timing analysis for ultra low-power CMOS VLSI," *Proc. of PATMOS'2002*, pp. 178-187, Sep. 2002.
- [6-9] K. Roy and S.C. Prasad, *Low-Power CMOS VLSI Circuit Design*, John wiley & Sons, Inc., 2000, pp 201-252.

- [6-10] T. Kobayashi and T. Sakurai, "Self adjusting threshold voltage scheme (SATS) for low voltage high speed operation," *IEEE CICC*, 1994, pp.271-.
- [6-11] S. Mutoh, "1-V Power supply high-speed digital circuit technology with multithreshold-voltage CMOS," *IEEE Journal of Solid-State Circuits*, vol. 30, pp. 847-, April 1992.
- [6-12] A. Fariborz, "A dynamic threshold voltage MOSFET (DTMOS) for ultra-low voltage operation," *IEDM Tech.*, 1994, pp809-.
- [6-13] L. Wei, Z. Chen, and K.Roy, "Double gate dynamic threshold voltage (DGDT) SOI MOSFETs for low power high performance designs," *IEEE SOI conference*, 1997, pp. 82-83.
- [6-14] S.S. Sapatnekar, V.B. Rao, P.M. Vaidya, and S Kang, "An exact solution to the transistor sizing problem for CMOS circuits using convex optimization," *IEEE Trans. On CAD of Integrated Circuits and Systems*, vol. 12, no. 11, pp. 1621-1634, September 1993.
- [6-15] T. Sakurai and A.R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas," *IEEE Journal Solid-State Circuits*, vol. 25, pp. 584-594, Apr. 1990.
- [6-16] T. Sakurai, "Closed-form expression for interconnect delay, coupling, and crosstalk in VLSI," *IEEE Transactions on Electron Devices*, vol. 40, pp. 118-124, Jan. 1993.
- [6-17] D. Sylvester and K. Keutzer, "Getting to the bottom of deep ubmicron," *Proc. of ICCAD*, pp. 203-211, 1998.

VITA

Kyu-Won Choi grew up in Seoul, Korea. He attended KyungHee University in Seoul, Korea, graduating with honors in 1991 with a Bachelor of Science degree in Electrical and Computer Engineering. In February of 1993, he received a Master of Science degree in Electrical and Computer Engineering from KyungHee University in Seoul, Korea. He subsequently joined Korea Telecom Research Laboratories in Seoul, Korea, where he worked on telecommunication-network protocol design.

Starting from the fall of 1996, he continued his education at the Georgia Institute of Technology, Atlanta, Georgia, where he conducted research on ultra low-power design for future generation systems. During his PhD. studies, he worked as an intern with Scientific Research Corporation in Atlanta, Georgia, from 1998 to 1999.

His research interests are in the area of wireless network protocols, low-power design techniques from software level to circuit/device level for future generation systems.