

**DATA MEMORY SUBSYSTEM RESILIENT TO PROCESS
VARIATIONS**

A Dissertation Presented

by

MAHMOUD BEN NASER

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2008

Electrical and Computer Engineering

© Copyright by Mahmoud Ben Naser 2008

All Rights Reserved

DATA MEMORY SUBSYSTEM RESILIENT TO PROCESS VARIATIONS

A Dissertation Presented

by

MAHMOUD BEN NASER

Approved as to style and content by:

Csaba Andras Moritz, Chair

Israel Koren, Member

Wayne P. Burleson, Member

Charles C. Weems, Member

Christopher V. Hollot, Department Chair
Electrical and Computer Engineering

*To my mother Amina, for her generosity, advice, and patience;
To my wife Taiba, for her endless love and encouragement;
To our beautiful children, Hessa and Abdullah, for filling my heart with joy
every day.*

ACKNOWLEDGMENTS

First of all, I am indebted to my advisor Prof. Csaba Andras Moritz for his encouragement, advice, mentoring, and research support throughout my whole Ph.D. study. Prof. Moritz's insights and ideas have provided me the greatest help to make this dissertation possible. I have found working with Prof. Moritz tremendously rewarding and truly feel like I have learned from the best!

I am appreciative and thankful to my committee members Profs. Koren, Burleson, and Weems for their valuable suggestions and comments, which led to a significant improvement in my research. They have modeled a lesson I will gladly carry forward with me in my future work with graduate and undergraduate students.

I want to thank Kuwait University for funding this research and providing a scholarship to pursue M.S. and Ph.D. degrees. A special thank you to all those whose support and friendship helped me to stay focused on this research and provided me with the encouragement to continue when the going got tough. Many thanks to Khaled Al-Barrak, Yao Guo, and Teng Wang.

My family deserves a great deal of credit for my development. I thank my brothers Mohammed, Abdurrahman, Abdulwahab, and Abdulaziz and my sisters Aisha, Khadeejah, Mariam, and Hafsa for all their love and support. Finally, I offer my most genuine thanks to my aunt Hessa and my cousin Altaf, whose unwavering compassion, faith, and love carry me through life.

ABSTRACT

DATA MEMORY SUBSYSTEM RESILIENT TO PROCESS VARIATIONS

FEBRUARY 2008

MAHMOUD BEN NASER

B.Sc., KUWAIT UNIVERSITY

M.Sc., BROWN UNIVERSITY

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Csaba Andras Moritz

As technology scales, more sophisticated fabrication processes cause variations in many different parameters in the device. These variations could severely affect the performance and power consumption of processors by making the latency of circuits less predictable and thus requiring conservative design approaches and/or techniques to increase performance that often affect power consumption. In this dissertation, we introduce and study step-by-step a 16KB cache subsystem in 32-nm CMOS technology, at both circuit and architecture levels, aiming for a single-cycle process-variation resilient subsystem design in a 1GHz processor that is high performance and power efficient at the same time. We use expected-case simulations in addition to worst-case circuit analysis to establish the overall delay and power consumption due to process variations under both typical and worst-case conditions. The distribution of the cache critical-path delay and

power consumption in the typical scenario was determined by performing Monte Carlo simulations at different supply voltages, threshold voltages, and transistor lengths on the complete cache design. In addition to establishing the delay and power variations, we introduce an adaptive variable-cycle-latency cache architecture that mitigates the impact of process variations on access latency by closely following the typical latency behavior rather than assuming a conservative worst-case design point, and allowing tradeoffs between power and performance to be controlled. We show that the proposed adaptive cache is transparent to other processor subsystems and has negligible power and area overhead compared to a conventional design. We also establish what the overall leakage power is due to process variations. The distribution of the cache leakage power was determined before and after incorporating state-of-the-art leakage optimizations. Simulation results show that our adaptive data cache is process-variation-resilient and can achieve on average 10% performance improvement on SPEC2000 applications in a superscalar processor, in conjunction with 6X reduction in the mean leakage power compared with a conservative design. Additional performance improvement potential exists in processors in which the data cache access is on the critical path, by allowing a more aggressive clock rate in the processor.

TABLE OF CONTENTS

	Page
ACKNOWLEDGMENTS	v
ABSTRACT	vi
LIST OF TABLES	xi
LIST OF FIGURES	xiii
 CHAPTER	
1. INTRODUCTION	1
1.1 Related Work	5
1.2 Contributions of Dissertation	7
1.3 Organization of Dissertation	9
2. LOW-POWER CACHE DESIGN	11
2.1 Introduction	11
2.2 Cache Organization Overview	12
2.3 Address Decoder Design	13
2.3.1 Circuit Style	13
2.3.2 Decoding Levels	15
2.3.3 Performance and Energy Evaluation	16
2.4 Tag Array Design	19
2.5 Data Array Design	25
2.5.1 Cache Subbanking	26
2.5.2 SRAM Cell Design	27
2.5.3 Divided/local Wordline	28
2.6 I/O Circuit Design	29
2.6.1 Cross-Coupled Inverter Latch (C^2IL)	30

2.6.2	Alpha Latch	30
2.6.3	Write Mechanism Design	34
2.6.4	Sharing Sense Amplifiers	34
2.6.5	Matchline Sense Amplifier	36
2.6.6	Control Circuits for Sense Amplifier Activation and Wordline Gating	37
2.7	Leakage Power Reduction Techniques	38
2.7.1	Dual- V_{th}	38
2.7.2	Stacked Transistors	39
2.8	Low-Power Baseline Cache	41
2.9	Chapter Summary	44
3.	PROCESS VARIATIONS IMPACT	45
3.1	Introduction	45
3.2	Worst-case Operating Conditions	47
3.2.1	Channel Length Variation	48
3.2.2	Threshold Voltage Variation	49
3.2.3	Supply Voltage Variation	51
3.3	Expected Behavioral Operating Conditions	52
3.4	Chapter Summary	53
4.	RESILIENT DATA MEMORY DESIGN.....	55
4.1	Introduction	55
4.2	Conservative Cache	55
4.3	Proposed Adaptive Cache	56
4.4	Mechanism and Implementation	58
4.5	Results and Analysis	61
4.5.1	Performance Speedup	62
4.5.2	Sensitivity to Issue Width	63
4.6	Chapter Summary	65
5.	POWER AND PERFORMANCE TRADEOFFS WITH PROCESS VARIATION RESILIENT ADAPTIVE CACHE ARCHITECTURES	67
5.1	Introduction	67
5.2	Impact of Process Variation in Caches	68

5.2.1	Cache Leakage Power	68
5.2.2	Impact of Channel Length Variation on Leakage Power	69
5.2.3	Impact of Threshold Voltage Variation on Leakage Power	71
5.2.4	Impact of Supply Voltage Variation on Leakage Power	73
5.2.5	Impact of All Parameters Combined	74
5.3	Leakage Reduction Technique	75
5.4	Adaptive Cache Architecture	78
5.5	Results and Analysis of Application Performance with an Adaptive Process Resilient Cache Architecture	81
5.6	Chapter Summary	83
6.	CONCLUSION	85
6.1	Future Work	86
	BIBLIOGRAPHY	89

LIST OF TABLES

Table	Page
2.1 6 X 64 Address Decoder Styles and Number of Stages	19
2.2 Tag Array Design	25
2.3 Cache Access Time for Different Number of Banks	26
2.4 Power Dissipation and Delay for Different Sense Amplifier Circuits	32
2.5 Total Cache Power Dissipation	35
2.6 Active Power Reduction Techniques	42
2.7 Leakage Power Reduction Techniques	42
3.1 Effect of Power Supply Voltage Variations	51
3.2 Nominal and 3σ Variation Values for Each Source of Process Variations Modeled	53
4.1 Overhead Associate with the Adaptive Design	62
4.2 Simple-scalar Parameters for CPU	63
5.1 Nominal and 3σ Variation Values for Each Source of Process Variations Modeled	69
5.2 Effect of Channel Length Variations	70
5.3 Effect of Threshold Voltage Variations	71
5.4 Effect of Power Supply Voltage Variations	73
5.5 Effect of Parameter Variations	75
5.6 Effect of Parameter Variations Using Leakage Enhancement Cells	77

5.7	Simple-scalar Parameters for CPU	81
5.8	Distribution of Cache Delay and Leakage Power for Different High- V_{th}	82

LIST OF FIGURES

Figure	Page
1.1 Application performance for different cache access cycles (see Table 5.7 for the ratio occurrence of these variations).....	3
2.1 Typical CAM-tag-based cache organization.	12
2.2 One-level decoders: (a) NOR; and (b) NAND based.	14
2.3 6-input dynamic (a) NOR; (b) NAND gate.	15
2.4 A 6X64 NAND decoder using three-bit predecoders.	17
2.5 Address decoders delay.	18
2.6 Address decoders power consumption.	18
2.7 Schematic of basic CAM cell (NCAM).	21
2.8 CAM cell with PMOS pull-down device (PCAM).	22
2.9 CAM cell circuitry with separate search bitlines from the write bitlines (CCAM).	23
2.10 Tag array design.	24
2.11 Total cache power as a function of the number of banks.	27
2.12 The structure of a basic 6T SRAM cell.	28
2.13 Data array with divided wordlines.	29
2.14 Cross-coupled inverter latch (C^2IL).	31
2.15 Alpha latch sense amplifier.	33
2.16 Write circuit.	34

2.17	8 to 1 column multiplexer.	35
2.18	Single-ended sense amplifier for match-line signal sensing.....	36
2.19	Control circuits for sense amplifiers and wordline gating.	37
2.20	Asymmetric SRAM cell [8].	39
2.21	Sense amplifier with stacked transistor.	40
2.22	Total cache power and delay.	43
2.23	Breakdown of power consumption in the optimized cache at T=75°C.....	43
3.1	Critical path of CAM-tag cache.	47
3.2	Effect of L_{eff} variation on cache delay.....	48
3.3	Effect of L_{eff} variation on power consumption.....	49
3.4	Effect of V_{th} variation on cache delay.....	50
3.5	Effect of V_{th} variation on power consumption.....	50
3.6	Distribution of the cache critical path delay.....	54
4.1	The proposed adaptive cache architecture (shown in a single five-stage pipeline).	57
4.2	The adaptive cache architecture during the classification phase.	59
4.3	The adaptive cache architecture during the execution phase.	60
4.4	BIST circuit.	61
4.5	Performance improvement between adaptive cache vs. a conservative cache using three-cycle access time.	64
4.6	Comparison of performance speedup for 16KB cache on four-way issue and eight-way issue SimpleScalar architecture.	64
5.1	The structure of a basic 6T SRAM cell.	69
5.2	Cache leakage power as a function of L_{eff}	70

5.3	Probability distribution of the cache leakage power as a function of L_{eff} .	71
5.4	Cache leakage power as a function of V_{th} .	72
5.5	Effect of V_{th} variation on cache leakage.	72
5.6	Cache leakage power as a function of V_{dd} .	73
5.7	Probability distribution of the cache leakage for different values of V_{dd} .	74
5.8	Distribution of the cache leakage power.	75
5.9	Leakage of enhanced SRAM cell [8].	76
5.10	Cache leakage power distribution using leakage enhancement cells.	77
5.11	Variable threshold-voltage scheme.	78
5.12	The proposed static adaptive cache architecture (shown in a single five-stage pipeline).	79
5.13	The static adaptive cache architecture during the execution phase.	80
5.14	Performance improvement between the adaptive caches vs. a conservative cache using three-cycle access time.	83
6.1	The dynamic adaptive cache architecture.	87

CHAPTER 1

INTRODUCTION

As technology scales, the feature size reduces, thereby requiring a sophisticated fabrication process. Parameter variations cause chip characteristics to deviate from the uniform, ideal values desired at design time. Three major sources of variation are often discussed: process variations, which consist of deviations in the manufactured properties of the chip, such as feature size, dopant density, etc.; voltage variations due to non-uniform power-supply distribution, switching activity, and IR drop; and temperature variations due to non-uniformities in the heat flux of different functional units under different workloads as well as the impact of non-uniformities in the chip's interface with its package [35].

The manufacturing process causes variations in many different parameters in the device, such as the effective channel length L_{eff} , the oxide thickness t_{ox} , and the threshold voltage V_{th} . These variations increase as the feature size reduces due to the difficulty of fabricating small structures consistently across a die or a wafer [10]. Controlling the variation in device parameters during fabrication is therefore becoming a great challenge for scaled technologies.

The performance and power consumption of integrated circuits can be greatly affected by these variations because they can make a given circuit exhibit different delays or power characteristics than intended during design. The process variations that are random in nature and are expected to become significant in the smaller geometry transistors are commonly used in memory [15].

The reasons for focusing on the caches are three-fold. First, caches consume a relatively large fraction of the processor area and power consumption. Second, level one caches have

to be built for minimum delay; hence, they tend to utilize low-threshold voltages. Finally, SRAM structures have a high number of independent critical paths and relatively low logic depth in those paths, which makes them very sensitive to process variations. Because of these three properties, the delay and power consumption of level one caches change significantly under process variations. Hence, the probability that a chip will not meet the performance/power constraints because of its cache is high [35].

In this thesis, we introduce and study step-by-step a 16KB cache, which exploits many low-power design techniques at the circuit and architectural levels. We target this cache design for a future 32-nm technology where leakage and process variation will play prominent roles [48]. We explore various design styles for each cache component and show that there is significant power consumed in the auxiliary circuitry, and careful design of all circuitry is necessary.

Moreover, we are interested in exploring both performance and power consumption issues related to process variations and gathering insights that could be used in new cache designs as well as possibly in developing new architectural techniques for fetch units and load-store units in microprocessors.

Since the operating frequency in microprocessors is typically determined by the expected delay of the slowest path, variation in the delay of the slowest path can make a single, fixed clock frequency too fast (causing errors) or too slow (incurring an opportunity cost). The question is whether there is significant delay variation overall that will drive a change in memory architecture design.

In state-of-the-art digital design, to ensure the correct execution in the memory circuit, the cache access delay must be decided based on the worst-case latency within the process variation range [35]. For example, even a small latency increase due to variation may require the whole cache access latency to be increased by one or more processor cycles. This can severely affect the overall application performance.

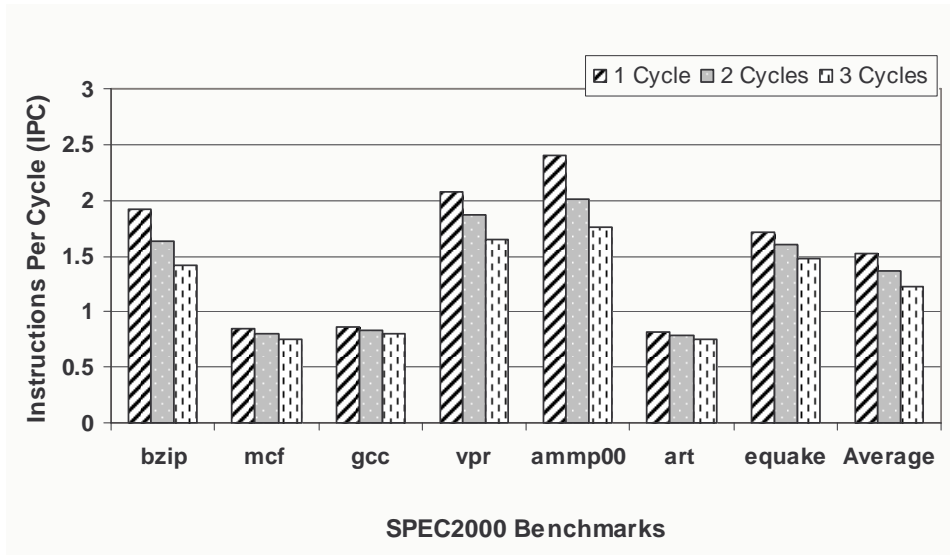


Figure 1.1. Application performance for different cache access cycles (see Table 5.7 for the ratio occurrence of these variations).

Preliminary simulation results with the HSPICE circuit simulator [2] show that process variations in effective channel length, supply voltage, and threshold voltage at 32-nm PTM device model [1] can affect the performance, after all factors are considered, at around 2-3X under the worst-case operating conditions [9]. To account for the worst-case scenario, we might need to increase the cache access time by two to three cycles or adopt other design approaches. Application performance could be impacted by as much as 30 to 40% as shown in Figure 1.1.

These results suggest that process variations must be taken into consideration while designing cache circuits and perhaps even architectures. There are several ideas that could be exploited in a memory system: 1) reduce performance by operating at a lower clock frequency (conservative approach); 2) increase cache access latency assuming worst-case delay (conservative approach); and 3) variable-delay cache architecture (adaptive approach proposed in this thesis). The first approach would clearly have a large impact on overall performance. The second approach would also have a significant impact, as shown above in Figure 1.1.

The goal of this dissertation is to estimate both the worst-case and typical-case delay variation expected in a state-of-the-art cache and to introduce an adaptive cache system that would mitigate the impact of process variations without taking any of the conservative design paths suggested earlier.

The proposed adaptive cache circuit is transparent to other subsystems and has negligible power and area overhead. Instead of accessing the cache with a fixed latency assuming worst-case latency, the proposed architecture can have different latency information stored in the delay storage unit for each cache line. The adaptive technique is enabled by a BIST circuitry, which tests the entire cache and detects the speed of the cache access. The speed results are then written into the delay storage. While we access each cache line, we will know the latency for that cache line, and the pipeline will act accordingly to save the waiting cycles. The expectation is that most of the cache lines will have much lower latency compared to the worst-case scenario.

Moreover, the power consumption of caches can be greatly affected by parameter variations and therefore are a major challenge for processor designers who need to meet power budgets and achieve desired performance levels consistently. We need to ensure that circuits and architectures are resilient to delay variations, without giving up much performance or consuming more power than nominally budgeted. Our adaptive design can control power and performance tradeoffs.

In addition to the performance and correction aspects, other objectives in this dissertation are (1) to analyze the effects of various parameter variations on the overall cache leakage power before and after applying leakage reduction optimizations; (2) to observe how leakage reduction techniques are affected by the variation: we investigate applications of leakage enhancement (LE) SRAM cells [8] to reduce leakage power; (3) what the expected leakage power distribution is in a complete cache; and (4) how an adaptive cache architecture would be able to control the tradeoffs between leakage and performance, including capturing effects at both circuit and application levels in a

superscalar microprocessor. We show that one can minimize power consumption while meeting the operating frequency constraints under the impact of process variations and with minimal effect on application performance compared to the optimal performance setting in the adaptive cache. Furthermore, we show that compared to a conservative design, an adaptive cache architecture resilient to process variations can even achieve an improvement in application performance in conjunction with a 6X lower leakage despite the effects of process variation.

1.1 Related Work

In recent years, process variation has been identified as one of the key threats to continued Moore's Law scaling. Bowman et al. performed some of the first analyses in the area of process variations and point out that a technology generation of performance can be lost due to process variations [13]. They also developed statistical models for process variations and their correlation with microarchitecture-driven parameters such as the number of independent critical paths and logic depth. Models were validated against real microprocessor chips fabricated in 0.25um and 0.13um process technology and found to be within 3% error. Mukhopadhyay et al. [52] proposed models for timing errors in SRAM memory due to random V_{th} variation. They considered several failure modes. Eisele et al. [31] have shown how random and systematic process variations affect overall performance of low-power designs.

At the circuit level, Narendra et al. [37] proposed forward body bias (FBB) to mitigate the impact of variation for several critical path structures. Tschanz et al. [64] proposed adaptive body bias and adaptive supply voltage techniques to reduce the variation in frequency of fabricated dies, improving the mean frequency and number of dies in the highest bin.

While there has been a lot of work on statistical methods to model and compensate for process variation at the circuit level [54] [5] [22] [13] [25], there has been little work

on modeling variations at the architecture level. Researchers have begun to explore the system-level impact of variations on reliability, performance, and power. Initial work in this area has focused on the modeling of process variations [58].

Several groups have proposed solutions to patch stability issues due to process variations in memory designs that use 6T SRAM cells [38] [55]. Kurdahi et al. [41] demonstrated analysis techniques to model and improve the yield of SRAMs at the system level by proper accounting for the coupling between the algorithms targeted for an SoC and the performance, power, and yield of SRAMs used in implementing them. These studies attack the yield loss problem at a lower level compared to our methods. Datta et al. [26] employed gate sizing to optimize individual stages in the processor pipeline optimizing yield for a given area constraint or minimizing area for a given yield constraint using the concept of area borrowing.

In [4], Agarwal et al. introduced a process-tolerant cache architecture to improve the yield in nanoscale memories. The proposed scheme detects and replaces faulty cells by dynamically resizing the cache. Another work by Datta et al. [27] tried to predict the yield of a pipelined circuit with analytical models. They observed that introducing imbalances among paths in a pipeline stage actually increases the yield of the design.

Furthermore, Humenay et al. [35] introduced a pre-RTL, architectural modeling methodology that incorporates the impact of process variations on multi-core chips, while Chandra et al. provided a methodology for modeling variations during system-level power analysis [18]. The effects of parameter variations and crosstalk noise on H-tree clock distribution networks are investigated in [20].

Perhaps the most relevant prior work is the “FMAX” model introduced by Bowman et al. [12]. FMAX is a predictive model for capturing the maximum frequency distribution of a microprocessor under the impact of process variations. However, all the simulation results in this dissertation are derived from detail circuit-level Monte Carlo simulations.

Researchers have shown that the selection of pipeline depth and other microarchitectural parameters at design time can significantly impact the susceptibility of an architecture to process variations. Borkar et al. conceptually demonstrated that the performance gain of deeper pipelines decreases due to the impact of within-die process variation [11]. Kim et al. performed a more quantitative analysis on pipeline depth under process variations [40].

Recently, Marculescu and Talpes proposed using globally asynchronous, locally synchronous (GALS) design techniques to design processors under process variations [47]. Liang and Brooks studied the impact of design-time microarchitectural tradeoffs on process variations [44].

Torrellas et al. [62] proposed the Dynamic Fine-Grain Body Biasing (DFGGB) technique to mitigate process variations, where different parts of the processor chip are dynamically given a voltage bias that changes the speed and leakage properties of their transistors, adapting to changes in operating conditions.

Variable-latency techniques have been proposed for register files and pipelined logic structures [43] [63] [50]. Liang et al. [45] proposed inserting level-sensitive latches inside the floating-point unit that can be enabled after fabrication. If process variation is such that the unit does not meet timing, the latches are enabled, adding one extra cycle to the floating-point unit.

In our work, we propose an adaptive variable-cycle-latency cache architecture that mitigates the impact of process variations on access latency. This dissertation presents a cache design that allows architects to reason how process variations affect it. Moreover, we establish what the overall leakage power is due to process variations in a cache and show how power and performance tradeoffs can be managed with the help of our adaptive cache subsystem despite process variation effects.

1.2 Contributions of Dissertation

This dissertation makes the following primary contributions:

- In this thesis, we introduce and study step-by-step a 16KB cache, which exploits many low-power design techniques at the circuit and architectural levels. We target this cache design for a future 32-nm technology in which leakage and process variation will play prominent roles. To model the power consumption in the memory system, we use state-of-the-art low-power cache circuits and simulate them using HSPICE.
- We explore various design styles for each cache component and show that significant power is consumed in the auxiliary circuitry, and careful design of all circuitry is necessary. Clearly, the choice of components and optimizations has a very significant impact on power consumption, which can easily be as much as 8X.
- In addition, we analyze the impact of different sources of process variation on cache delay and power consumption. An improved critical path design is used to determine cache delay distribution.
- Monte Carlo simulations were used in addition to worst-case circuit analysis to establish the overall delay due to process variations in a cache subsystem under typical case conditions. The distribution of delay of a cache critical path was determined by performing Monte Carlo sampling at different supply voltages, threshold voltages, and transistor lengths.
- We propose an adaptive variable-cycle-latency cache architecture that mitigates the impact of process variations on access latency by closely following the typical latency behavior rather than assuming a conservative worst-case design point.
- We show how characteristics of the modern processor architecture (e.g., superscalar, out-of-order execution) can mitigate the impact of process variations while offering significant performance and power advantages. We have implemented the proposed adaptive cache architecture in SimpleScalar [14] and simulated a set of SPEC2000 [3]

benchmarks to compare the performance. Our results show that the new architecture can achieve a 9% to 21% (on average, 17%) performance improvement on the applications studied compared to a conventional design assuming worst-case latency, while providing resilience against failures due to process variations.

- In our analysis, the area overhead associated with the extra hardware (BIST, delay storage, and control circuitry) has been evaluated by using the Synopsys Design Compiler tool for the adaptive cache.
- We establish what the overall leakage power is due to process variations in a cache and show how power and performance tradeoffs can be managed with the help of our adaptive cache subsystem despite process variation effects.
- The distribution of the cache leakage power was determined by performing Monte Carlo simulations at different sources of process variation on adaptive cache design before and after incorporating leakage optimizations. Simulation results show that our static adaptive data cache is process-variation-resilient and can achieve on average 10% performance improvement on SPEC2000 applications in a superscalar processor, in conjunction with 6X reduction in the mean leakage power compared with a conservative design. In addition to performance benefits, the proposed cache can thus achieve large power savings.

1.3 Organization of Dissertation

This dissertation is composed of the following chapters in addition to this introductory chapter:

In Chapter 2, we introduce and study step-by-step a 16KB cache in 32-nm PTM technology [1], which exploits many low-power design techniques at the circuit and architectural levels. We explore various design styles for each cache component and show that significant power is consumed in the auxiliary circuitry, and careful design of all

circuitry is necessary. In addition, we present a case study of the overall cache power when all the low-power techniques discussed are applied and suitable components selected.

Chapter 3 presents a detailed analysis of the impacts of process variation on this cache delay and power consumption under worst-case and expected behavior conditions. To estimate the typical delay in a cache, we determine the distribution of delays by performing Monte Carlo sampling at different supply voltages, threshold voltages, and transistor lengths.

Chapter 4 describes a new architecture technique to mitigate the effect of process variations and proposes a variable-cycle adaptive cache. We have implemented the cache at the circuit level and extended the SimpleScalar architecture simulator [14]. We use a set of SPEC2000 [3] benchmarks to compare the performance of the adaptive cache with a conventional approach.

Chapter 5 presents a detailed analysis of the impact of process variations on caches focusing on power consumption. To estimate the leakage power in a cache, we determine the distribution of leakage power by performing Monte Carlo simulations for different sources of process variation on adaptive cache design. In addition, we describe new static adaptive cache architecture techniques to mitigate the effect of process variations and reduce power consumptions.

Finally, a summary chapter includes a summary of the results in this dissertation and a number of directions for future work in this area.

CHAPTER 2

LOW-POWER CACHE DESIGN

2.1 Introduction

In the last decade, power/energy consumption has become a critical design criterion in microprocessors. The memory system, including caches, consumes a significant fraction of the total system power. For example, the caches and translation look-aside buffers (TLB) combined consume 23% of the total power in the Alpha 21264 [33], an ARM10TDMI design would consume more than 50% of its energy in the caches and memory management units in 0.13um TSMC at 400MHz, and the caches alone use 42% of the power in the StrongARM 110 [51]. Developing energy-efficient and high-performance memory systems is a key challenge for next-generation processor design.

While there has been much of research on approaches to reduce the total power consumed in caches, there is still plenty of confusion surrounding questions such as the following: (1) how much power is consumed in auxiliary cache circuits, (2) what is the fraction of power consumed in different parts when all components are using state-of-the-art low-power circuits, and (3) how do different design styles affect overall cache power and delay?

In this chapter, we go through a design of a 16KB cache, which exploits many low-power design techniques at the circuit level. We explore various design styles for each cache component and show that significant power is consumed in the auxiliary circuitry, and careful design of all circuitry is necessary.

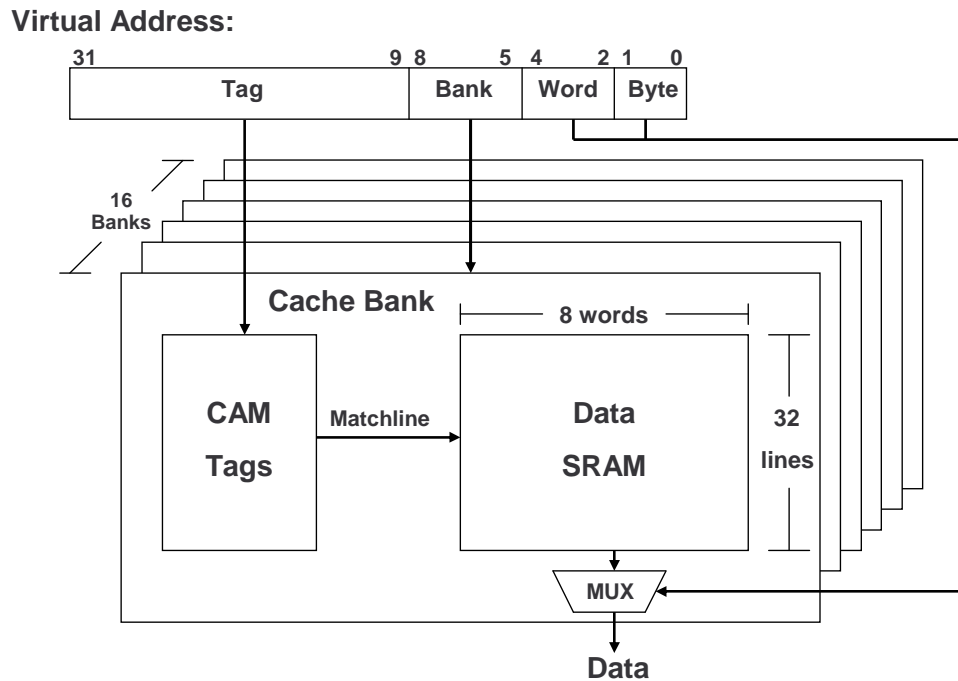


Figure 2.1. Typical CAM-tag-based cache organization.

2.2 Cache Organization Overview

The focus of this dissertation is on CAM-based caches (see Figure 2.1). This choice is motivated by many recent designs in industry [32] and academia [67] that show CAM caches to be often preferred in low-power processors. CAM caches are supported in set associative and fully associative configurations.

Key components of a cache include the tag and data arrays, address and bank decoders, and auxiliary circuitry. The main source of power consumption in caches is the power dissipated by: bitlines such as due to precharging, read and write cycles (tag and data arrays), wordlines when a particular row is being read or written, tag comparisons, peripheral circuitry such as sense amplifiers, control logic to switch on/off sense amplifiers, and various address/bank decoders.

Virtual addressing is assumed in this thesis for the cache. This is often more energy efficient as such a design does not require an address translation for every cache access [7]

to be completed before the tag access, and it is preferred in many embedded applications (especially single-application devices without context switching) and CAM-based designs (our choice).

Since the address space is much larger than the capacity of the cache, only a small portion of the address is used to map a given block to a location in the cache. Figure 2.1 shows how the 32-bit address space is used. The cache is divided into sixteen banks of 1KB each. Each bank contains 32 lines, and the low-order bits of the address are used to select one of the banks, each of which is fully associative internally.

2.3 Address Decoder Design

In general, the address decoder is the starting point of the cache access; it is a combinational circuit with n input address lines and 2^n output lines. Each of the 2^n possible input combinations activates exactly one of the output wordlines.

Since the access time of the first-level on-chip cache is in the critical path of the pipeline, the address (or bank) decoder can have a substantial impact on the speed and power consumption of the cache.

In a CAM-based design, the bank decode is on the critical path for a regular access, while the (cache line) address decoder is not as it is used only for writes during replacement. In low-power designs, one could also use a 3x8 local wordline decoder to select only the word in a cache line for each match.

The decoder design problem is three-fold: determining the optimal decoder circuit, finding the optimal number of stages, and determining the size for each level.

2.3.1 Circuit Style

The decoder function can be implemented via different combinations of NAND, NOR, and Inverter gates. Let us consider the design of a 6-to-64 address decoder. Each of the

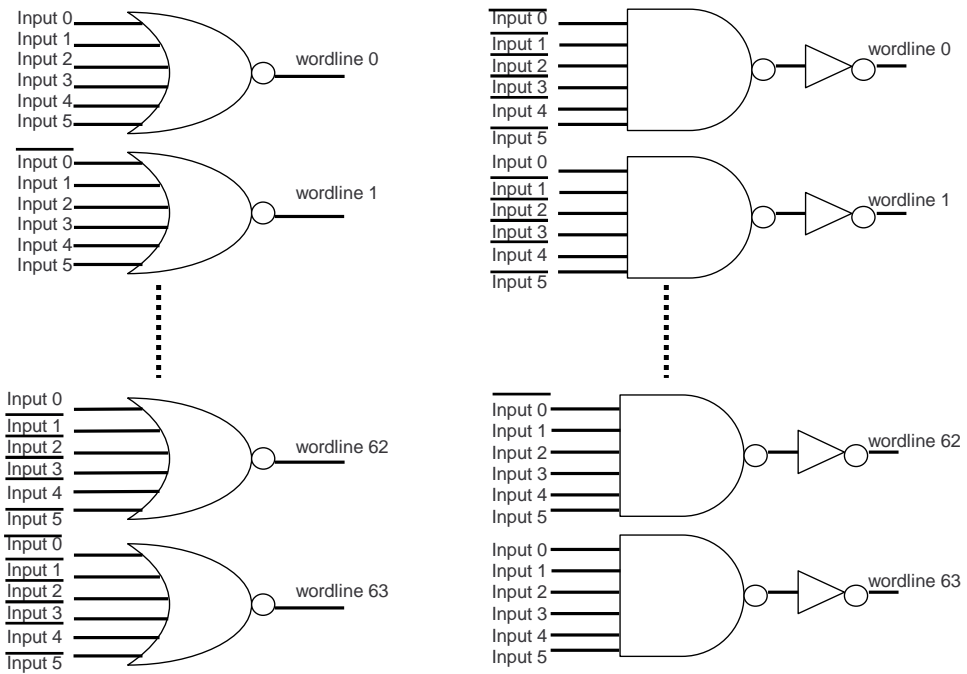


Figure 2.2. One-level decoders: (a) NOR; and (b) NAND based.

wordline outputs WL_i is a logic function of the 6-input address signals ($Input_0$ to $Input_5$). For example, the row with address 63 is enabled by the following logic function:

$$WL_{63} = Input_0 * Input_1 * Input_2 * Input_3 * Input_4 * Input_5$$

Alternatively, using Inverting logic the logic function is:

$$WL_{63} = \overline{Input_0 + \overline{Input_1} + Input_2 + \overline{Input_3} + Input_4 + \overline{Input_5}}$$

As shown in Figure 2.2, the decoder can be implemented by using a 6-input NOR decoder or a 6-input NAND decoder. NOR decoders are substantially faster, but consume more area and power than their NAND counterparts. As is clear from the following observation, only a single wordline is being pulled down after the precharge phase in a NAND decoder, while only a single wordline stays high in the NOR case. In addition, each gate can be implemented by using a different logic style: dynamic logic, static logic, or a combination.

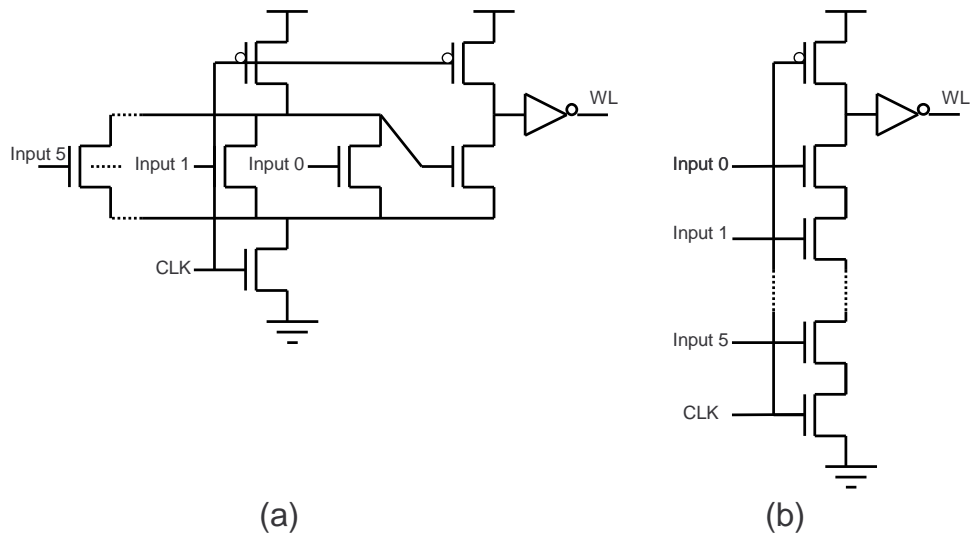


Figure 2.3. 6-input dynamic (a) NOR; (b) NAND gate.

Static logic has the advantage of being robust in the presence of noise and has no static power dissipation. However, for complex gates with large fan-in, static logic becomes expensive in terms of performance and area.

On the other hand, dynamic logic makes it possible to implement fast and small complex gates, at the expense of reduced robustness with regards to noise. It is sensitive to parasitic effects such as leakage, charge redistribution, and clock feed through. Figure 2.3 shows an example of a dynamic logic based on a 6-input NOR gate and a 6-input NAND gate.

2.3.2 Decoding Levels

The propagation delay of the address/bank decoder is often important, since it can add directly to both the read and the write access times. The main problem is that a large decoder will require a very large number of transistors. The capacitance associated with the long runs of wires and high gate-input count will add to the delay. The address inputs will also have to be buffered to drive this huge capacitance load. Another problem is that the power consumption of such a decoder will be very high due to the large number of gates.

The concept of predecoders is used to solve these problems, which decodes segments of the address in a first logic layer. A second layer of logic gates then produces the wordline signals. Consider the case of the 6-input NAND decoder. The expression for WL_{63} can be regrouped in the following way:

$$WL_{63} = \overline{\overline{(Input_0 * Input_1 * Input_2)} + \overline{(Input_3 * Input_4 * Input_5)}}$$

For this case, the address is partitioned into sections of three bits that are decoded in advance. Then the resulting signals are combined using 2-input NOR gates to produce the fully decoded array of wordline signals. The resulting structure is shown in Figure 2.4.

The use of a predecoding stage is clearly advantageous for address decoders as it reduces the fan-in to the NAND gate and the fan-out of the address inputs, yielding significant performance gain. This configuration also has another advantage: Adding a select signal to each of the predecoders makes it possible to disable the decoder when a memory block is not selected. This gives important power savings.

2.3.3 Performance and Energy Evaluation

The Cadence tool was used to design the address decoder at the schematic level, and HSPICE simulation was used to evaluate the performance and power dissipation of different decoder designs.

All the decoders were designed using 32-nm technology and simulated with a supply voltage of 0.9V. All the gates in the decoder are minimum sized. Reducing performance to the lowest possible amount allows for more gains in power efficiency. Table 2.1 shows number of stages for each decoder style. Figure 2.5 and Figure 2.6 summarize the results obtained showing the delay and power consumption. As we expected, the 6-input NOR decoder shows the best performance results and the worst power consumption. The simulations show an 82% average power savings from using the 3-input Dynamic NAND-2-input Static NOR decoder over the 6-input NOR decoder. The saving is obtained without a significant degradation in performance.

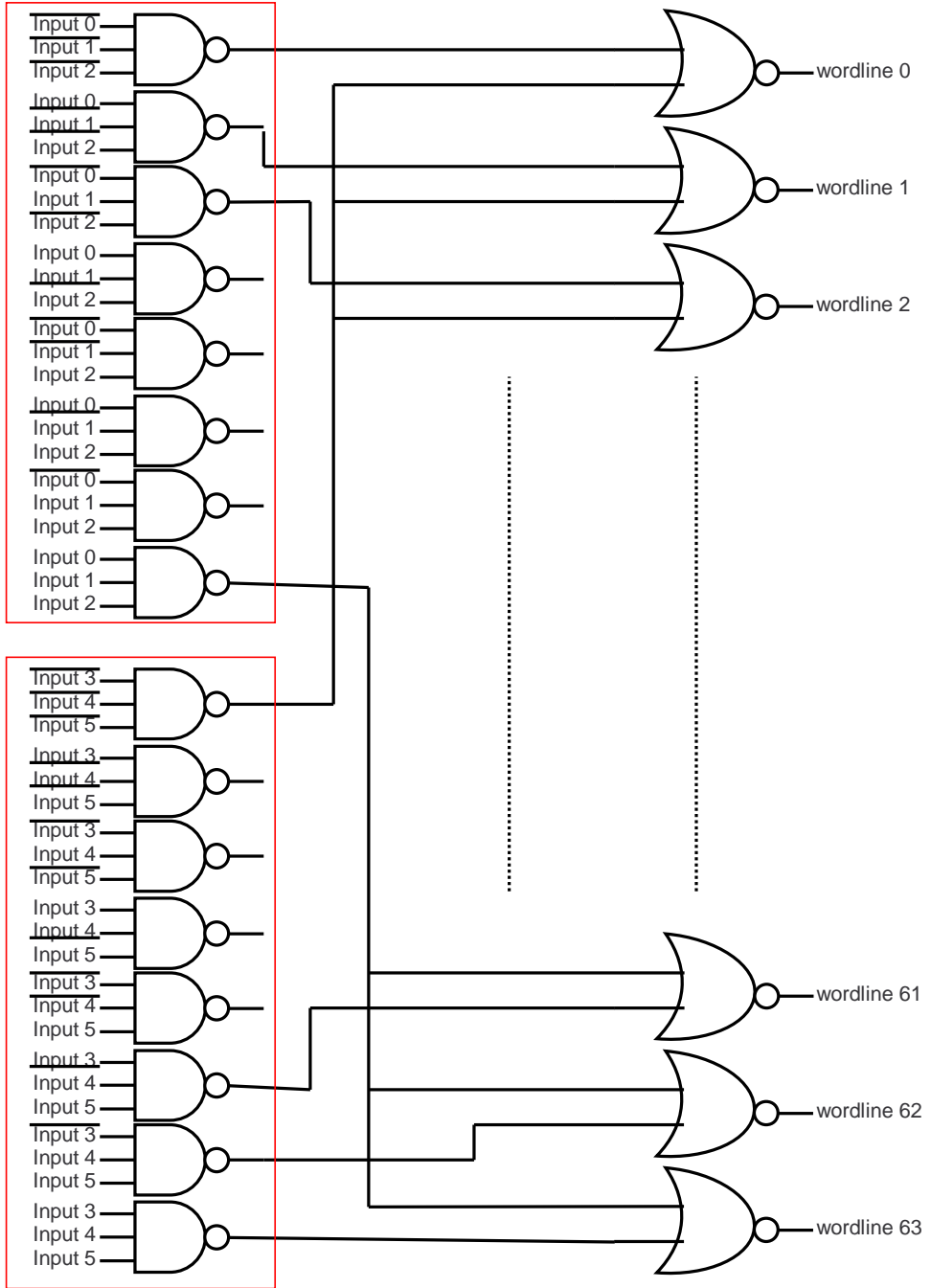


Figure 2.4. A 6X64 NAND decoder using three-bit predecoders.

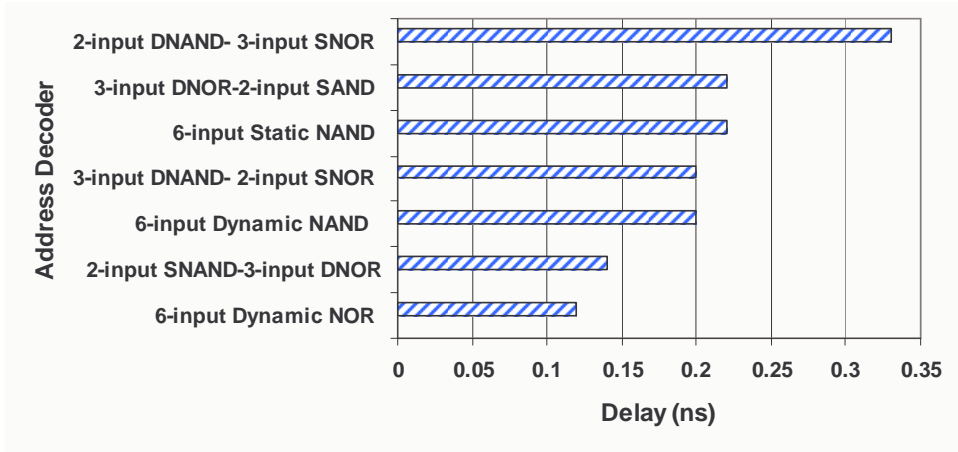


Figure 2.5. Address decoders delay.

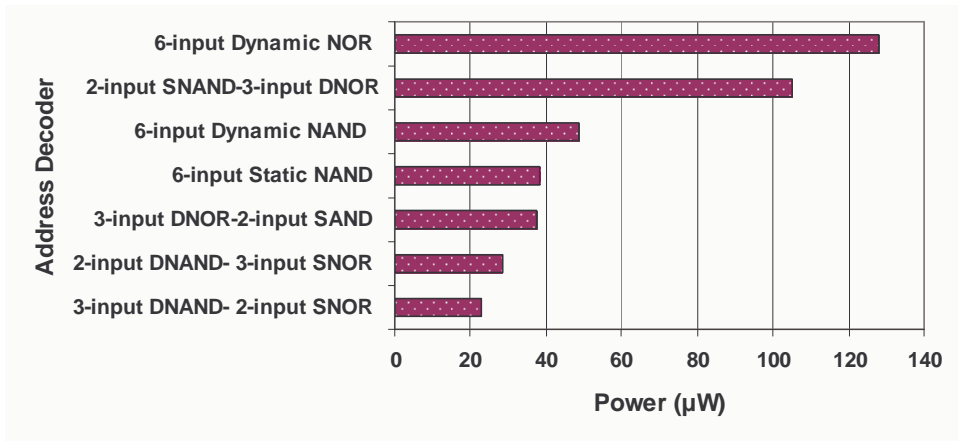


Figure 2.6. Address decoders power consumption.

Table 2.1. 6 X 64 Address Decoder Styles and Number of Stages

Decoder Style	# of stages
6-input Dynamic NOR	1
2-input SNAND-3-input DNOR	2
3-input DNOR-2-input SAND	2
2-input DNAND- 3-input SNOR	2
6-input Dynamic NAND	1
3-input DNAND- 2-input SNOR	2
6-input Static NAND	1

The split decoder approach yields reduced energy and good delay comparatively. Which design to select depends upon the primary design requirement: speed, area, power consumption, ease of design, or robustness. As will be discussed in Section 2.8, for our cache design we selected a 4-input static NOR (bank decoder) and a 3-input DNAND-2-input Static NOR gates (address decoder used during replacement). A 3-input static NAND decoder is used for the local wordline decoder.

2.4 Tag Array Design

Content addressable memory (CAM) provides a unique exclusive fast data search function by accessing data by their content rather than their memory location. Depending on how the tags are stored and organized, a cache may be classified as CAM-based or SRAM-based. A CAM-based cache stores tags in a CAM array while a SRAM-based cache stores tags in a static RAM array. Both types of caches store data in SRAM arrays.

Direct-mapped caches, although simpler, can be inefficient in terms of energy usage because they experience more misses due to conflicts. Many modern processors employ set-associative caches as L1 or L2 caches. Since an n -way set-associative cache has n locations where a cache line can be placed, it can offer higher hit rates than direct-mapped caches.

However, increasing cache associativity makes the cache access time longer due to the delay in the way selection is based on tag-comparison results. In addition, this method is quite wasteful of power because at least $n-1$ tag and data reads will be wasted for each cache access. To avoid this disadvantage, several researchers have proposed CAM-tag caches [30], [67]. The CAM-tag cache attempts to avoid the unnecessary way activation by looking up the tag before the data are accessed.

CAMs perform tag checks in parallel. If a CAM match is found (i.e., on a cache hit), the cache provides the word data in the associated cache line to the processor (for read). Otherwise, a cache-line replacement takes place. Another advantage of CAM-tag caches is that they simplify the handling of cache stores. Cache wordlines are enabled only on cache hits; stores can therefore happen in a single cycle. A further advantage is the high associativity possible with CAM designs and schemes where individual cache lines can be locked.

In fact, CAM-tag caches have been employed in many low-power processors. For example, ARM3 [24] has a 4KB 64-way set-associative CAM-based cache, ARM10 uses a 32KB banked cache, and the Intel XScale processor [32] employs 64-way set-associative CAM tags. Zhang and Asanovic have shown in [67] that CAM-based caches have comparable access latency, but yield lower hit energy and higher hit rates than SRAM-based set-associative caches at the expense of approximately 10% area overhead. In this thesis, we have evaluated three CAM cell designs:

In [65], Sodini et al. proposed a 5-transistor dynamic CAM cells suitable for high-density arrays and capable of storing three states “0,” “1,” and “don’t care,” which is useful for many applications (e.g., image processing). The dynamic CAM cell has a disadvantage that it requires cell refresh to preserve the content of the CAM cell due to the impact of leakage current. A dynamic CAM cell is not suitable for a low-power cache.

The first basic CAM cell is based on a static memory cell (Figure 2.7). Data are stored in the two cross-coupled inverters. The two NOMS transistors controlled by the wordline

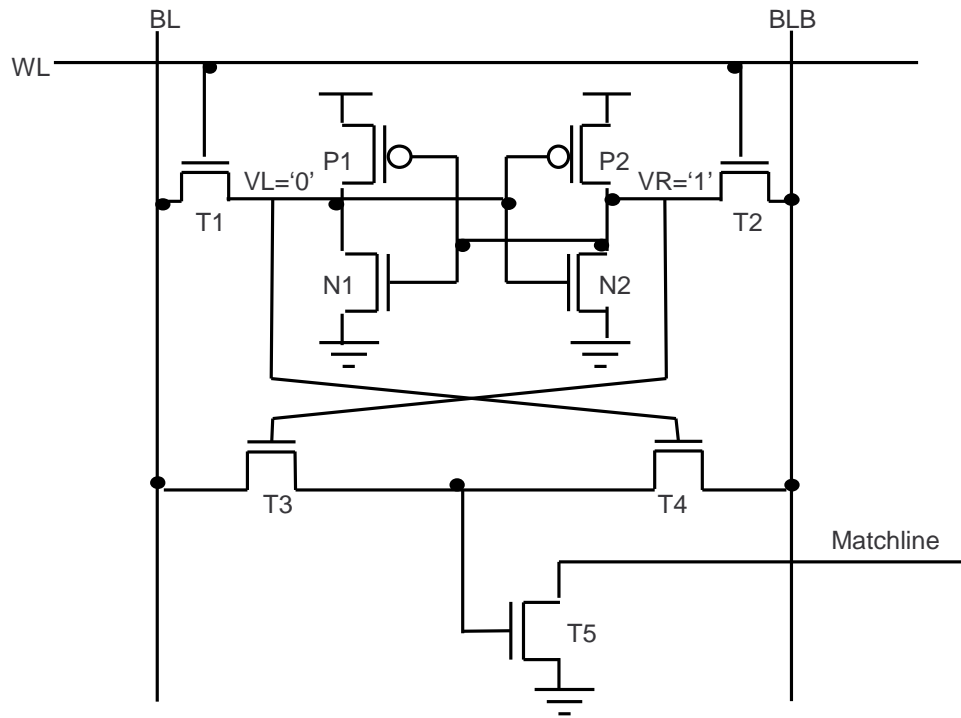


Figure 2.7. Schematic of basic CAM cell (NCAM).

(T1 and T2) allow the CAM cell to be written. Only one of the two transistors (T3 and T4) will be activated at a time, since the gates are connected to two opposite sides of the memory cell. If the search line matches the value in the memory cell, then transistor T5 will turn off, creating no path to ground for the match line [59].

The second CAM cell design evaluated in this work is a nine-transistor cell as shown in Figure 2.8. The cell incorporates an ordinary six-transistor SRAM cell to store a bit of data, a comparison circuit containing two NMOSs, and a PMOS word matchline driver device [34].

A major difference from the conventional nine-transistor CAM cell with active pull-down, shown in Figure 2.7, is that the pull-down device to drive the match line to a low level is a PMOS rather than an NMOS transistor. The comparator devices gate nodes are connected to the storage nodes in such a way that the match-line driver PMOS is off when a

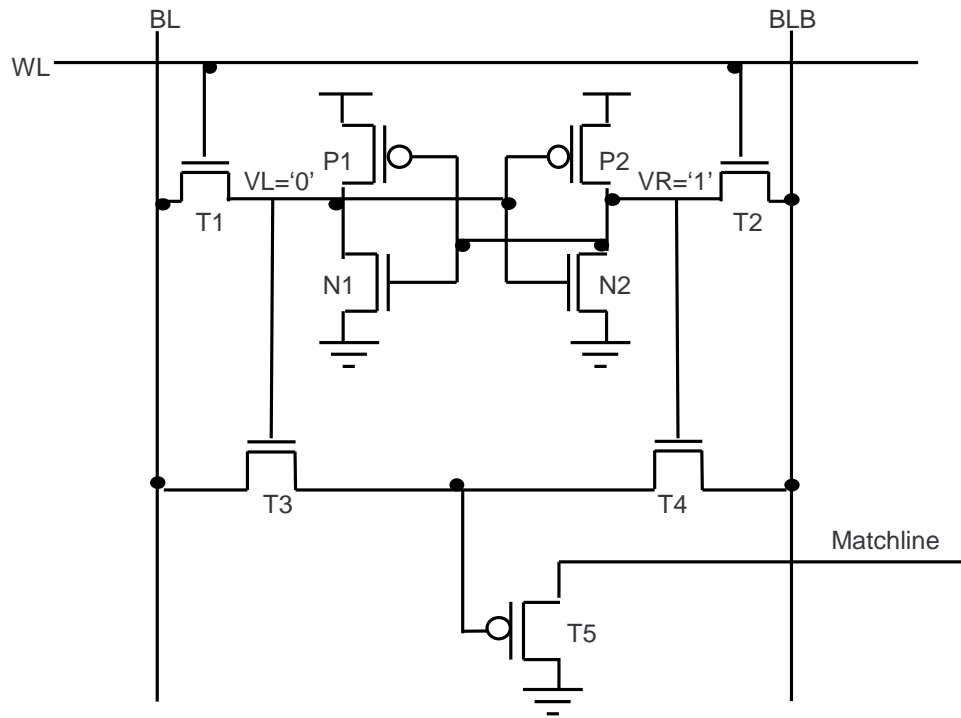


Figure 2.8. CAM cell with PMOS pull-down device (PCAM).

match is detected in the cell, which is the same as with the conventional NMOS pull-down device cell.

Advantages brought by having a PMOS instead of an NMOS as the match-line pull-down device are as follows: the match-line discharge level is a level elevated by the threshold voltage of the deeply back-gate-biased PMOS, not a full low level or the ground level, leading to smaller rush current, smaller capacitive coupling noise, and power savings through voltage swing reduction [34].

Figure 2.9 shows the third CAM cell design we evaluate in this thesis [67]. The cell contains a SRAM cell and a dynamic XOR gate used for comparison. The match line is precharged high and conditionally discharged when there is no match.

A major difference from a conventional nine-transistor CAM cell is that the search bitlines are separated from the write bitlines, in order to reduce the capacitance switched during a search operation. Figure 2.10 shows the diagram for the tag array design.

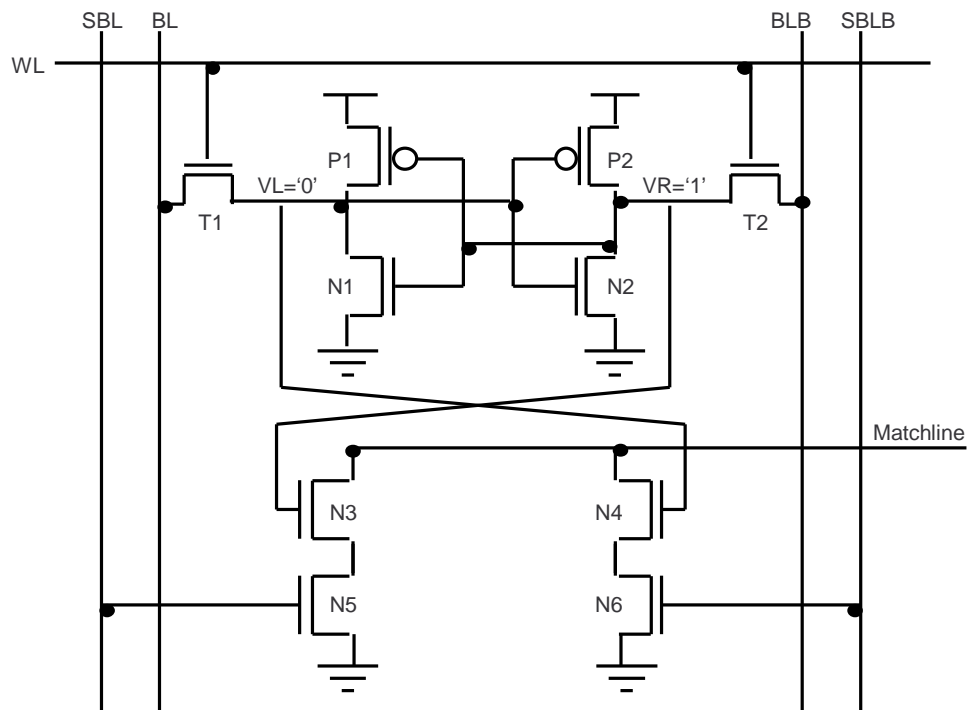


Figure 2.9. CAM cell circuitry with separate search bitlines from the write bitlines (CCAM).

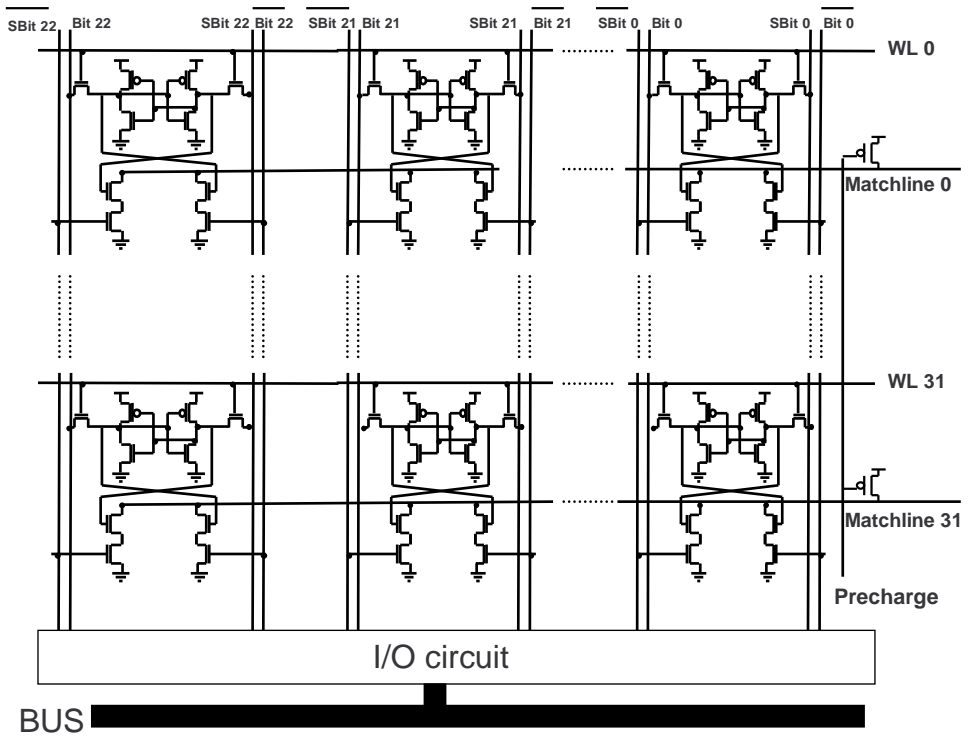


Figure 2.10. Tag array design.

The power of the CAM array is the sum of the power in the match lines and search lines. Since there is at most one match per access, all but one of the match lines discharge. To measure the power consumption of the three CAM cells, a tag array of 32 rows was used with each row of 23 CAM cells and single-ended sense amplifier circuit.

All the CAMs were designed using 32-nm PTM technology [1] and simulated using HSIPCE. Three different CAM cells of different sizes were implemented. The minimum transistor size for each CAM cell was determined through circuit simulations. The optimum ratio was searched by varying the channel widths and lengths of the transistors. Table 5.7 summarizes our results showing delay, average power consumption, and area for each CAM design.

The CCAM cell proves to be more power efficient and faster than the other two designs (see Table 5.7), because the search bitlines are separated from the write bitlines in order to reduce the capacitance switched during a search operation. Nevertheless, the difference is minimal at this design point that we attribute to the relatively short bitline height in our banked design.

Table 2.2. Tag Array Design

CAM Design	Delay(ns)	Power (mW)	Area (μm^2)
NCAM	0.436	5.21	2297.54
PCAM	0.427	5.17	2503.29
CCAM	0.403	5.14	4242.18

2.5 Data Array Design

Crucial in reducing the power dissipation of a cache is the data array design. Following are a few methods that have been used effectively to reduce cache data array power.

2.5.1 Cache Subbanking

The largest capacitive elements in memory are the wordlines and bitlines, each with a number of cells connected to them. Therefore, reducing the size of these lines can reduce power consumption.

A common technique often used in large caches is called *Cache Subbanking*. In cache subbanking, the data array is partitioned into subbanks. Only the memory cells in the activated bank have their bitlines driven.

CAM-based caches use some of the low-order bits of the address to select one of the banks. Within each bank, a fully associative search is used. The associativity depends on the memory bank and cache line sizes.

For a fixed size cache design, the amount of power savings depends on the number of banks. In this section, we evaluate the performance and the power consumption of 16KB cache using HSPICE simulation at 32-nm PTM technology.

From Table 3.1, there are two effects we observe when we increase the number of banks. First, the individual bank access time decreases because each bank is smaller. Second, the time spent on getting the address bits to the bank and routing the data out of the memory increases because we have to travel a greater distance. The benefits from the reduced bank access time outweigh the increase in routing time, and so we obtain overall reduced total cache access times.

Table 2.3. Cache Access Time for Different Number of Banks

Number of Banks	1	2	4	8	16	32	64
Delay(ns)	1.26	0.99	0.88	0.83	0.81	0.78	0.77

Figure 2.11 shows the total power consumed by the cache as we increase the number of banks at T=75°C. The different columns are for a different number of banks. The basic observation is that as the number of banks increases the dynamic power consumed by the

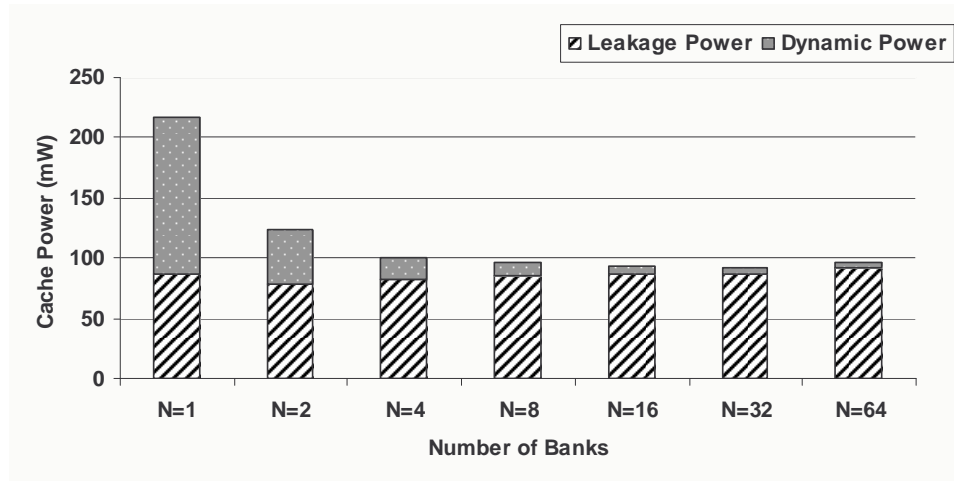


Figure 2.11. Total cache power as a function of the number of banks.

cache decreases. The increase in the number of banks means a greater number of output drivers, sense amplifiers, etc., and consequently an increase in the leakage power.

Cache subbanking significantly reduces the total power consumption of cache. Since the CAM-tag array is still consuming power for each cache access, the amount of power savings by a subbank depends not only on the number of banks in a cache but also on the cache size. Subbanking is typically very effective and can be implemented with minimal logic cost. In our 16KB design, we have found a 1KB bank (see Figure 2.11) size to maximize the benefits from the technique.

2.5.2 SRAM Cell Design

Static Random Access Memory (SRAM) is the memory block that holds the data. The size of the SRAM array determines the size of the cache. Low-power cache designs typically use a six-transistor (6T) SRAM cell instead of a four-transistor (4T) cell primarily due to the DC current always drawn by a 4T cell design.

A 6T SRAM cell is designed as a full-CMOS cross-coupled inverter circuit, with each inverter using one NMOS and one PMOS transistor. A 4T cell replaces the PMOS

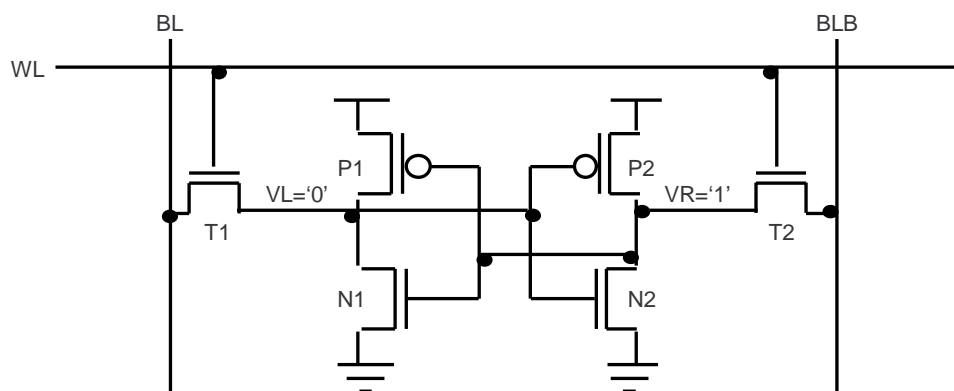


Figure 2.12. The structure of a basic 6T SRAM cell.

transistors with resistors, such that one of the transistors is always drawing current whether the cell is holding a zero or one value.

We have decided to use a basic 6T SRAM cell, as shown in Figure 2.12. The wordline control signal allows the cell to be accessed for reading or writing and shuts off access to the cell otherwise. The isolation transistors also protect the value stored in the cell during precharging phases.

2.5.3 Divided/local Wordline

The main idea of this technique is to divide each row of SRAM cells into segments, and use a local decoder to access only one segment. Only the memory cells in the activated segment have their bitlines driven [66]. The goal of the divided wordline approach is to reduce the delay and active energy of the SRAM wordline by reducing total capacitance. Unlike cache subbanking, which achieves a similar goal by breaking up a tall and wide array into smaller sub-arrays, the divided wordline approach achieves its goal by reducing the number of access transistors connected to the wordline. This improves performance (by decreasing wordline delay) and lowers the power consumption (by decreasing the number of bitline pairs activated). However, local decoders increase delay and area. A design based on a divided wordline is shown in Figure 2.13.

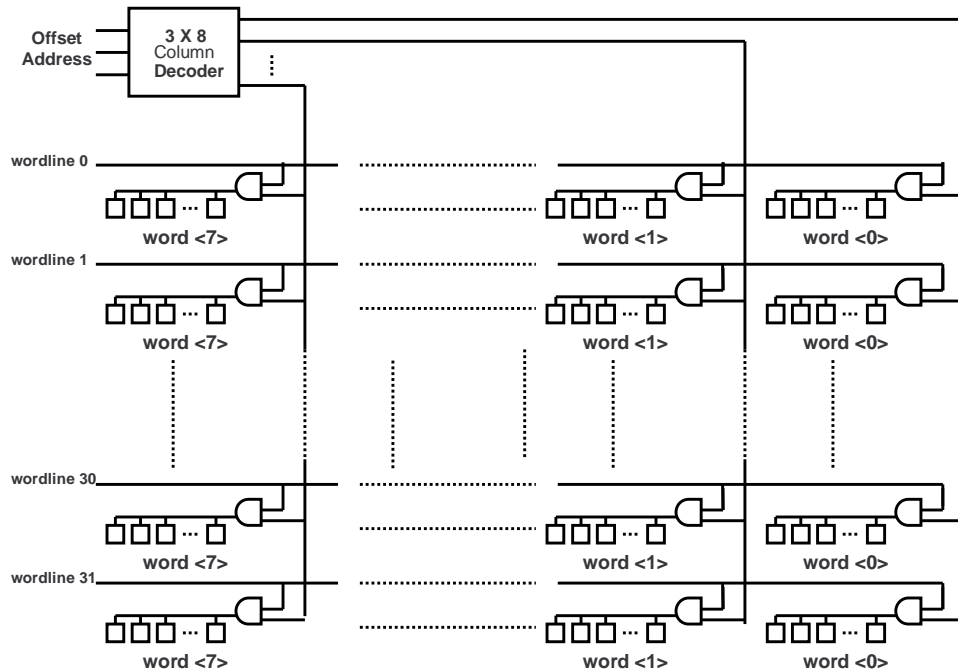


Figure 2.13. Data array with divided wordlines.

To evaluate the divided wordline technique, we apply it to a 16KB 32-way set-associative cache with 32-byte cache lines. Our power measurements show that the overall cache power consumption is reduced by 11% with the divided wordline technique.

2.6 I/O Circuit Design

A key component in the periphery of a cache is the sense amplifier. Each sense amplifier monitors a pair of bitlines and detects when one changes. By detecting which line goes low, the sense amplifier can determine what is in the memory cell. Since the sense amplifier is in the critical path of the memory access, its performance strongly affects both cache access time and overall cache power dissipation.

There is a wide variety in the types of sense amplifier circuits used in caches, for example, current mode sense amplifiers and voltage mode sense amplifiers. A current mode sense amplifier converts and amplifies a small difference in current to a full-rail voltage at

the output node. The voltage sense amplifier senses a small change in voltage that is a result of a particular cell being switching onto a bitline.

From the results in [53], the current sense amplifiers have better performance than the voltage sense amplifiers. This is due to the low-impedance termination of the bitlines, which allows them to be almost independent of the bitline capacitance. However, the current sense amplifiers have higher power dissipation than the voltage sense amplifiers. This disadvantage is due to the direct path current that exists in the current sense amplifiers.

For our low-power cache design, we have tested two different voltage sense amplifiers: a cross-coupled inverter latch and the alpha latch. For other designs, the reader is referred to [53].

2.6.1 Cross-Coupled Inverter Latch (C^2IL)

Figure 2.14 shows the schematic of the cross-coupled inverter latch. It is based on two cross-coupled inverters across the bitlines. The sense amplifier nodes are precharged and equalized. When the read circuit is enabled, these cross-coupled inverters sense the difference in voltage between the bitlines. The output of the inverter goes to “1” or “0” depending on the voltage level sensed.

The disadvantage of this sense amplifier is that the bitlines are connected directly to the sense amplifier’s output node, thereby affecting the swing in the bit-lines. In addition, this design is sensitive to erroneous latching from device mismatches or timing errors.

2.6.2 Alpha Latch

The alpha latch is similar to C^2IL except that the bitlines are terminated at the gates of two NMOS input devices and not the output nodes of the sense amplifier. Figure 2.15 shows the schematic of the alpha latch. The advantages of alpha latch sense amplifiers are as follows:

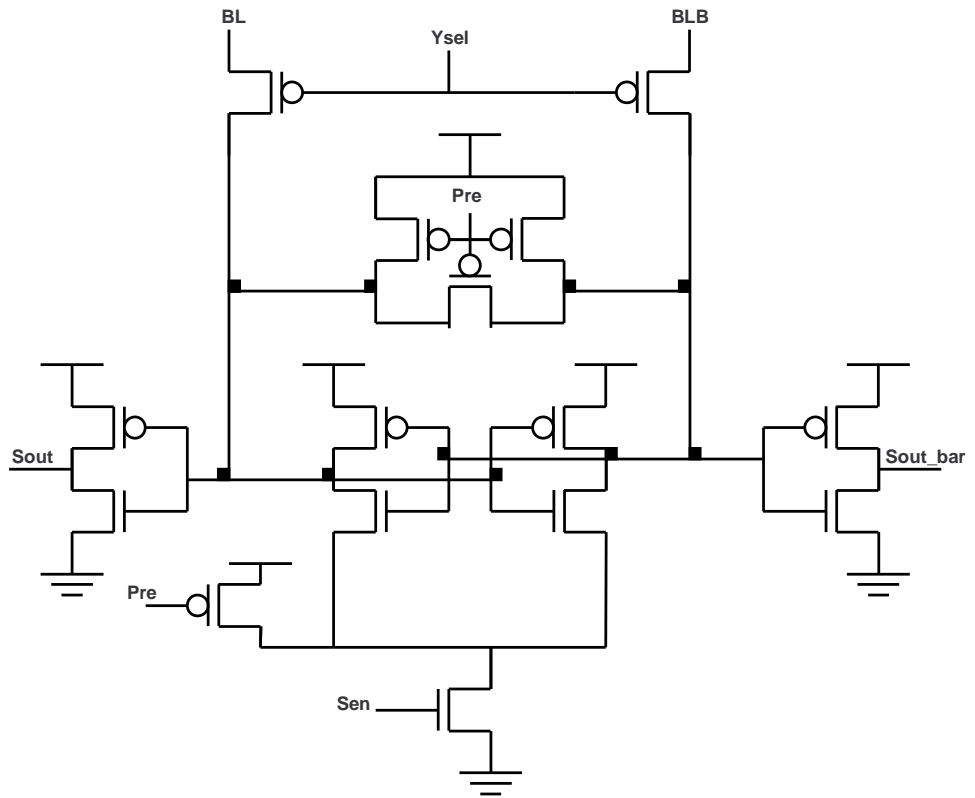


Figure 2.14. Cross-coupled inverter latch (C²IL).

- high sensitivity and high-speed amplification because of the separation of inputs and outputs and the lightly loaded buffered outputs, in addition to the inherent high performance resulting from the circuit structure;
- no direct-current flow from the power supply line to the ground line except during the state transition period, which is very short;
- a latching function, which holds the amplified signal even after the input signals have disappeared as long as the input nodes are at the voltage levels at which the input stage devices are ON and the sense amplifier is kept enabled.

To measure the power consumption of the sense amplifiers circuits, a cache bank with 32 rows of 24 CAM cells and 256 columns of SRAM cells was used with sense amplifier circuits. The threshold voltages for the NMOS and PMOS transistors were 0.2V and -0.2V, respectively. The simulations were conducted under different operating temperatures.

In Table 3.2, it can be seen that the cross-coupled inverter latch is a little faster than the alpha latch. This is because the two additional transistors in series in the alpha latch make it slower than C^2IL .

Table 2.4. Power Dissipation and Delay for Different Sense Amplifier Circuits

Sense Amplifier Circuit	C^2IL	Alpha Latch
Temperature	75°C	
Delay (ns)	0.78	0.81
Power (mW)	9.83	7.28
Temperature	100°C	
Delay (ns)	0.80	0.83
Power (mW)	11.76	8.71

The power measurements show that the alpha latch proves to be the best candidate for a low-power embedded design. The reason is that the alpha latch isolates the output node from the bitlines, thereby providing low swing in the bitlines as opposed to C^2IL . In fact,

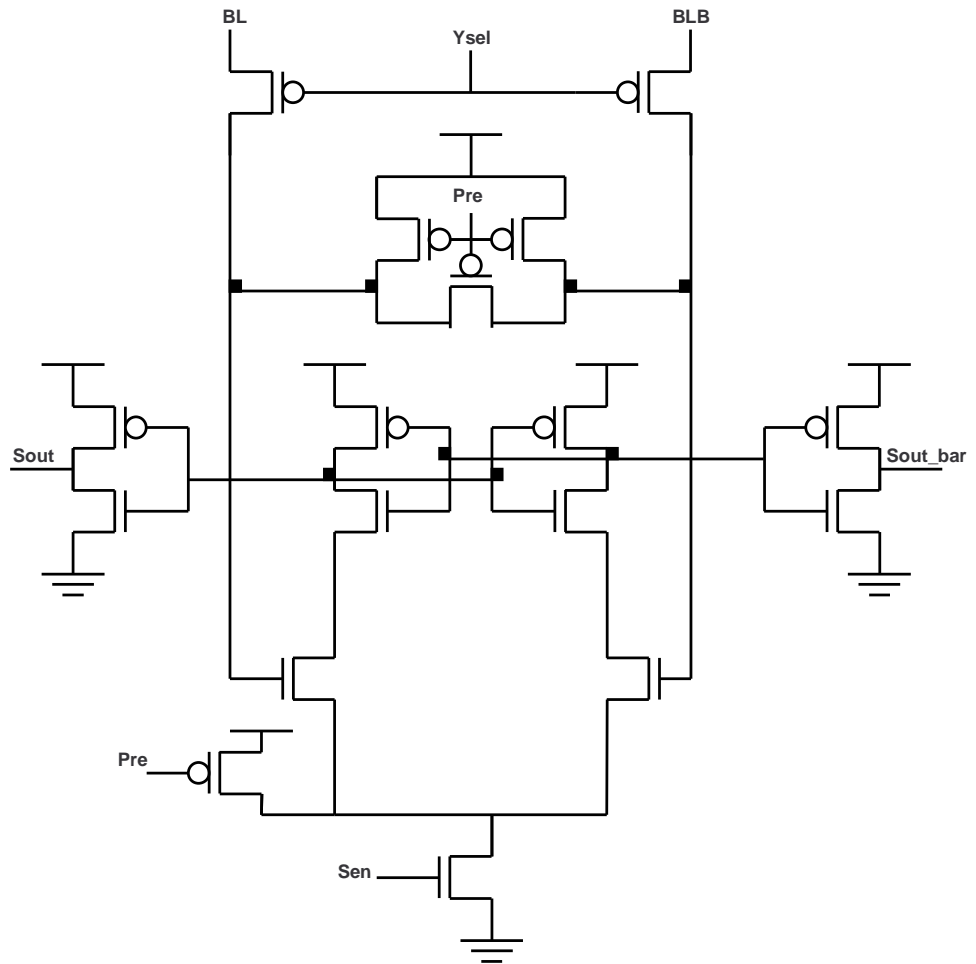


Figure 2.15. Alpha latch sense amplifier.

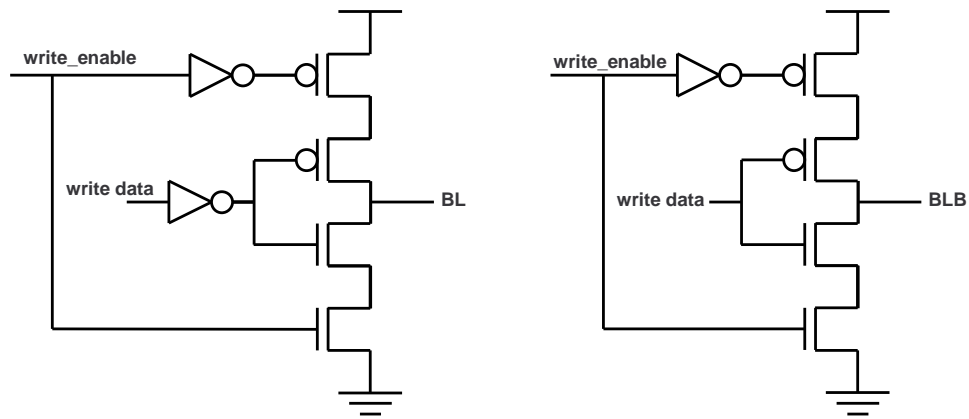


Figure 2.16. Write circuit.

we have noticed that the advantages of the alpha latch are further accentuated once leakage power is accounted for in a 32-nm design.

2.6.3 Write Mechanism Design

To write a value to a cell, the write circuitry has to force the value into the cell and has to be able to overpower the transistors around the cell. This is very critical, especially when a “0” has to be written. At the same time, when a read cycle is taking place or the circuit is simply not enabled, the write circuitry has to be completely cut off from the other processes. This call for a tri-state device, which is enabled by write enable and write data to allow the data to be written (Figure 2.16).

2.6.4 Sharing Sense Amplifiers

It is possible for one sense amplifier to be shared among several pairs of bitlines. In this case, a multiplexer is inserted before the sense amps; the select lines of the multiplexer are driven by the decoder.

In many caches, a column select multiplexer is used to reduce the number of sense amplifiers required. Figure 2.17 shows such a multiplexer. A single sense amplifier is

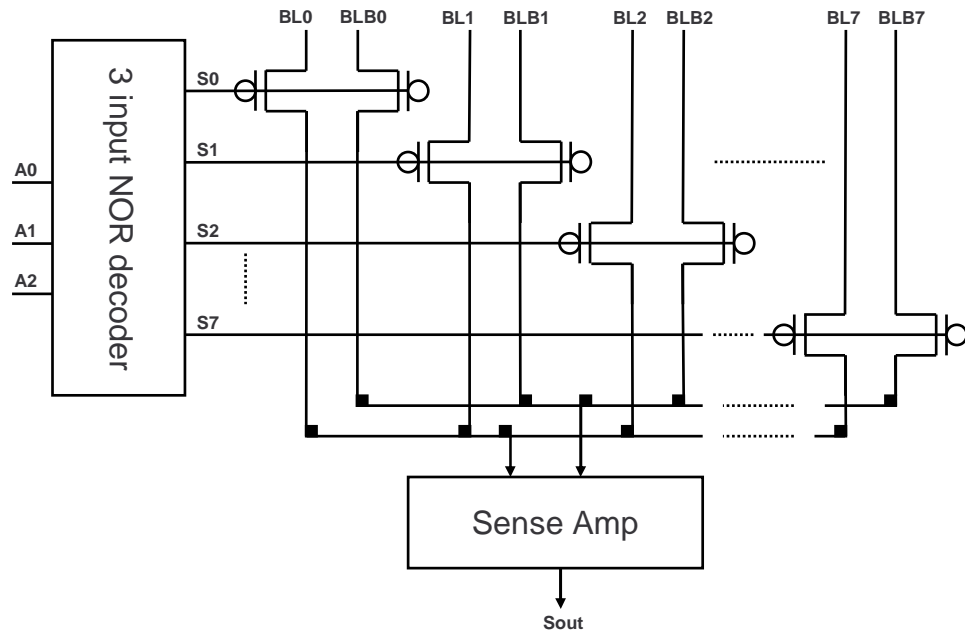


Figure 2.17. 8 to 1 column multiplexer.

shared among eight pairs of bitlines. The gate of the pass transistor is driven by signals from the output of the decoder.

For example, a 32-byte cache line can be partitioned into eight words in which there are 32 sense amplifiers shared among 256 pairs of bitlines. Table 2.5 shows the impact of sharing the sense amplifiers on the total cache power consumption. Our power measurements show that the power savings from sharing sense amplifiers (we have evaluated the alpha latch) can be as high as 8 to 11% of the total cache power. This savings is primarily from the reduction in leakage.

Table 2.5. Total Cache Power Dissipation

Temperature	50°C	75°C	100°C
Using Sharing S. A.	16.6 mW	27.6 mW	46.5 mW
Without Sharing S. A.	18.1 mW	30.7 mW	52.3 mW

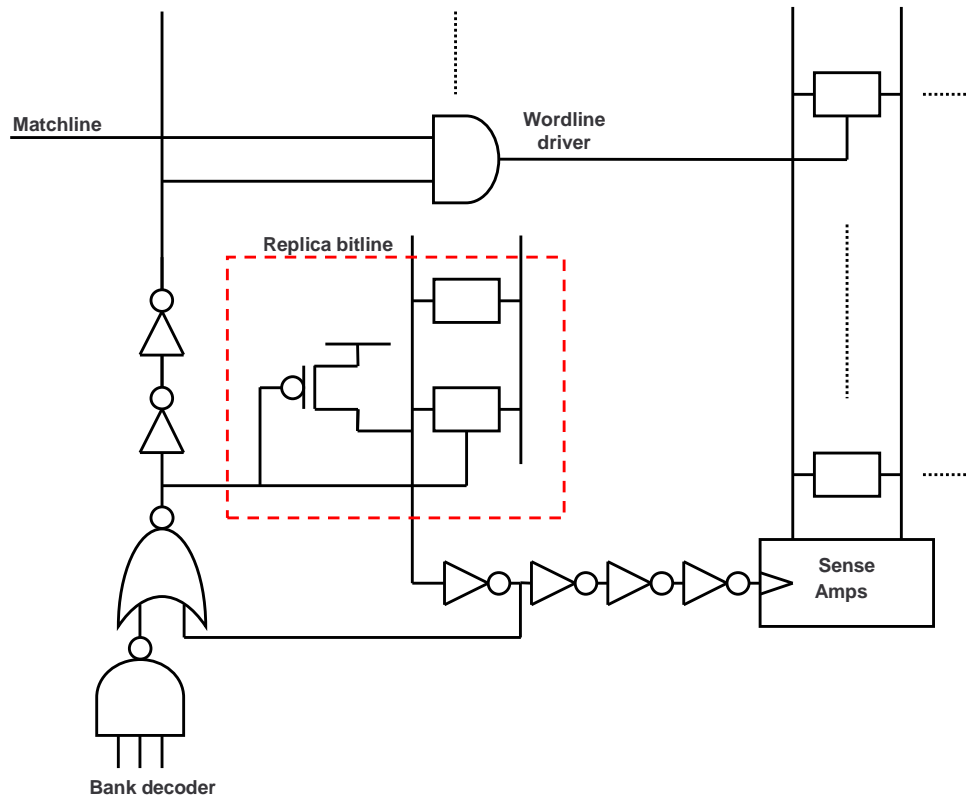


Figure 2.19. Control circuits for sense amplifiers and wordline gating.

2.6.6 Control Circuits for Sense Amplifier Activation and Wordline Gating

In order to complete our cache design, we used a replica circuit to control the sense clock and wordline gating, as shown in Figure 2.19. The circuit helps to minimize the variations in the speed and power of SRAMs across varying operating conditions [6].

Replica memory cells and bitlines are used to create a reference signal whose delay tracks that of the bitlines. This signal is used to generate the sense clock with minimal slack time and control wordline gating to limit bitline swings. First, the bank decoder activates. Then, the sense clock fires exactly when the minimum bitline swing has developed; the minimum bitline swing for correct sensing is around 15% of the supply.

2.7 Leakage Power Reduction Techniques

Microprocessor performance has been improved by increasing the capacity of on-chip caches. However, the performance gain comes at the price of increased power consumption. Power can broadly be divided into two components, active power and leakage power. Active power is the power consumed by switching parts of the digital circuits. Leakage power is the power consumed by the transistors when they are off. Leakage has increased exponentially with newer technology generations. This exponential increase is caused by the fundamental scaling of device dimensions and threshold voltage. Leakage power of a chip is expected to increase much more rapidly than the increase in active power [28].

There are different mechanisms that contribute to leakage power. These include subthreshold leakage, reverse-biased PN junctions, punchthrough currents, hot carrier effect, gate leakage, BTBT leakage, etc. However, the main contributor of leakage power is the subthreshold leakage current [46]. Subthreshold leakage is the current that flows from the drain to the source even when the transistor is off.

Several research and industrial groups have introduced circuit and architectural techniques to reduce leakage power. Our contributions in this section are to examine the power/performance tradeoffs of these techniques applied to the caches. We present two leakage-reduction techniques.

2.7.1 Dual- V_{th}

One solution for lowering subthreshold leakage currents, named dual- V_{th} , uses a mix of transistors tailored to the circuits function: high-performance transistors on the critical path and low leakage transistors in areas that have more slack for delay [19].

The majority of the SRAM cells spend their time in the standby state. In this state, most of the leakage power is dissipated by the transistors that are off and have a voltage differential across their drain and source.

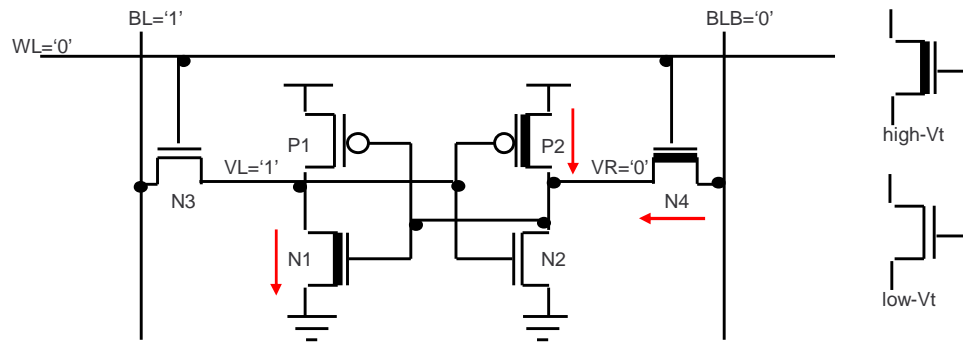


Figure 2.20. Asymmetric SRAM cell [8].

This technique could be implemented in a SRAM cell (called asymmetric cell). In this technique, the leaking transistors are replaced with a high- V_{th} transistor, where all other transistors are replaced with regular- V_{th} ones [8]. The technique allows reduced leakage current while preserving memory state.

In the asymmetric cell in Figure 2.20, the leakage power of this cell is comparable to the high- V_{th} cell in the preferred state and comparable to the low- V_{th} cell otherwise. This scheme offers leakage reduction with no overhead in area.

2.7.2 Stacked Transistors

Another circuit technique to reduce subthreshold leakage current adds a low-leakage transistor between a circuit and the power or ground connection (or both) [56]. This technique is named stacked transistor to describe the additional transistor that acts as a gate, opening and closing a connection to the power supplies.

Figure 2.21 shows a schematic diagram of the stacked transistor technique applied to a sense amplifier; in this example, an NMOS gating transistor is placed between the sense amplifier and ground. As shown in the diagram, the sleep signal controls the leakage mode of the circuit.

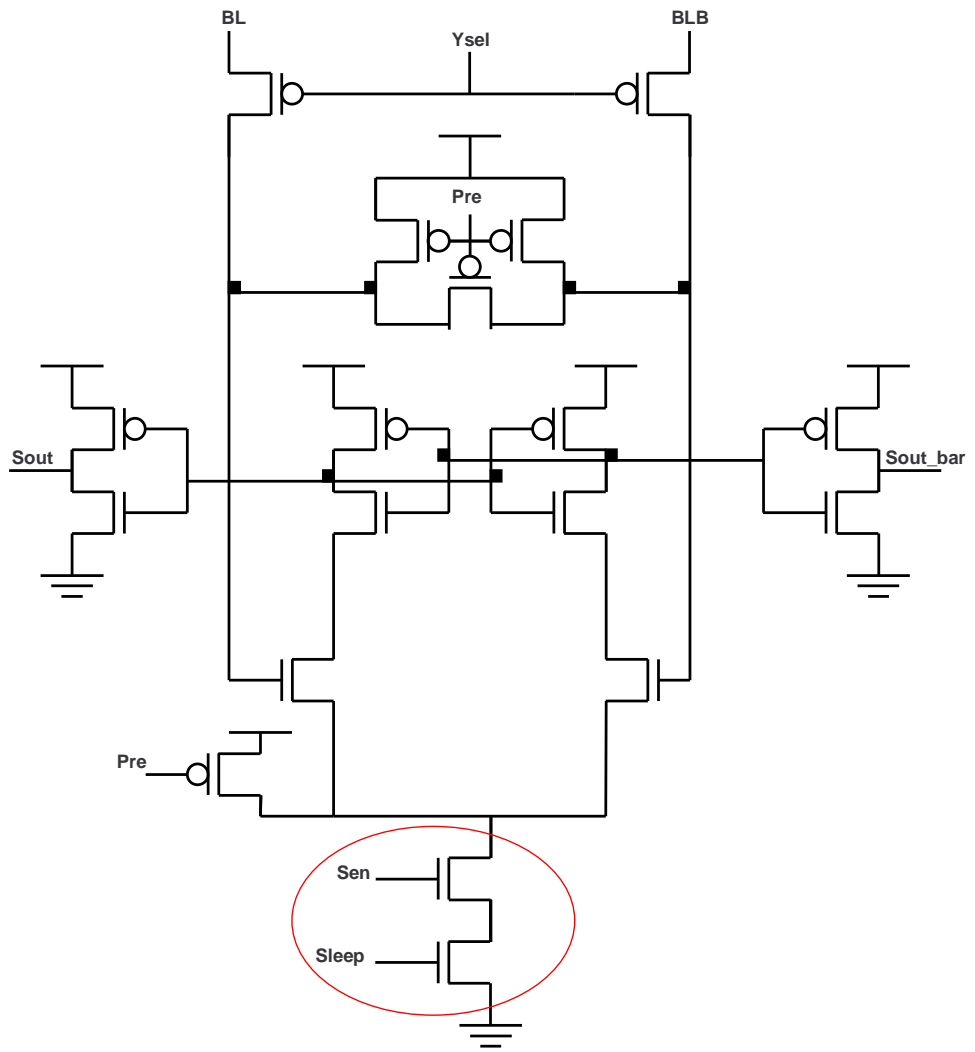


Figure 2.21. Sense amplifier with stacked transistor.

While the sleep signal is asserted, the circuit is connected to the ground and functions as a standard sense amplifier. To place the circuit in idle mode, the sleep signal is desasserted, turning the transistor off and interrupting the current path through the circuit.

Leakage current is reduced when the gating transistor disconnects sub-circuits from power supplies, reducing the leakage power dissipated by the circuit and reducing energy consumption throughout the duration the circuit is in the low-leakage idle mode.

2.8 Low-Power Baseline Cache

This section presents a case study of the overall cache power when all the low-power techniques discussed are applied and suitable components selected. First, we introduce an initial cache with no dedicated power optimization techniques.

The initial cache design is a 16KB, uses nine-transistor CAM cell (NCAM cell) and the 6T SRAM cells, for the tag array and the data array circuits, respectively. A single-ended sense amplifier was used for match-line signal sensing in the tag array, and a cross-coupled inverter latch was used for read data sensing in the data array. We do not use sharing sense amplifiers and wordline gating in the initial cache design. For the bank decoder, we select a 4-input static NOR gates based design; for the cache line decoder, we apply two-level decoding: first, level 3-input dynamic NAND and, second, level 2-input static NOR gate.

Table 2.6 shows the cache design with low-power optimization techniques added. Table 2.7 shows the leakage reduction techniques. The cache was designed using 32-nm technology and simulated with a supply voltage of 0.9V and a clock of 1GHz. The simulations were done in the HSPICE circuit simulator using the BSIM4.0 model. The dynamic power was measured for a single word cache read. A 5pi wire model was used. The wire parameters were obtained from Predictive Technology Model for interconnect [1].

Figure 2.22 shows the delay and power consumption in this cache before and after applying all the low-power techniques to the initial cache design. By applying many state-of-the-art low-power techniques, at the circuit level alone, the cache power consumption

Table 2.6. Active Power Reduction Techniques

Cache Component	Power Optimization Technique
Bank Decoder	4-input static NOR gates
Tag Array	10-transistor CAM Cells
Data Array	6T SRAM Cells
Cache Line	Wordline Gating
Line Decoder	Two level decoding: First level 3-input Dynamic NAND gate and Second level 2-input static NOR gate
Tag and Data Arrays	Cache Subbanking (16 banks)
Bank size	1KB
Sense Amplifiers	Alpha latch sense amplifier and sharing sense amplifiers

Table 2.7. Leakage Power Reduction Techniques

Cache Circuit	Leakage Technique
Data Array Sense Amplifiers	Stacked transistor
Tag Array Sense Amplifiers	Stacked transistor
SRAM Cell	Asymmetric SRAM cell
CAM Cell	Stacked transistor
Wordline Gating	High- V_{th}

is reduced by 8X for the same speed and functionality. Improving the power efficiency of caches is critical to overall system power efficiency.

A component-wise power consumption breakdown is necessary to evaluate the actual effectiveness of these power reduction techniques. Figure 2.23 shows the percentages of the power consumption for each component in the cache after applying all low-power optimization techniques. The key power source is the CAM cells power consumption.

The performance requirement in the 1GHz cache design limits the leakage reduction. Slower design points could have a much larger fraction of the leakage component reduced. The SRAM cells power can also be further reduced with architectural techniques, e.g., leakage can be optimized further in idle banks (Drowsy Cache [36]). What we also

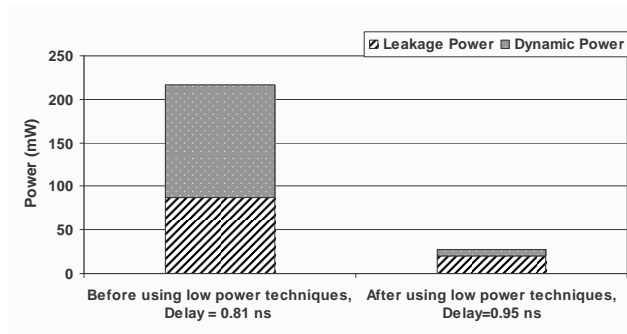


Figure 2.22. Total cache power and delay.

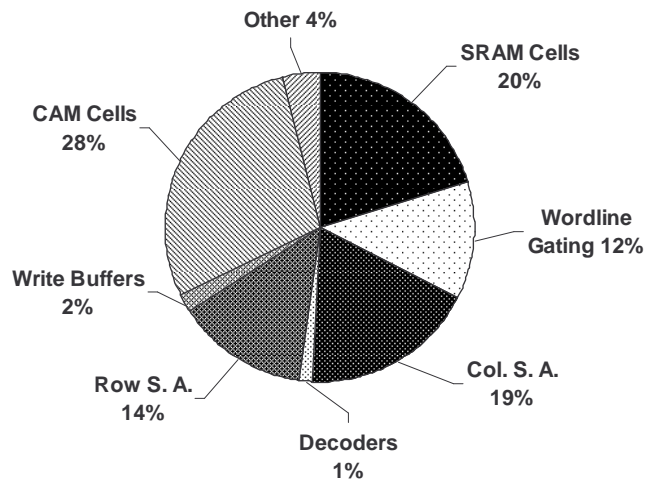


Figure 2.23. Breakdown of power consumption in the optimized cache at T=75°C.

observed is that as the performance requirement goes up, the leakage in the SRAM cells becomes dominant compared to other sources of power.

2.9 Chapter Summary

In this chapter, we explored and evaluated complete cache design including all auxiliary circuitry in deep submicron technology. A comparative study of different cache components using 32-nm technology has been presented with a target clock rate of 1GHz. We evaluated both active and leakage power and showed how different parts of a cache affect overall power efficiency and delay. Energy reduction techniques have been evaluated in the Address Decoders, CAM-based tags, SRAM data arrays, and all I/O circuitry. By applying many state-of-the-art low-power techniques, at the circuit level alone, the cache power consumption is reduced by 8X for the same speed and functionality. Improving the power efficiency of caches is critical to the overall system power efficiency. Our contributions in this chapter are to examine the power/performance tradeoffs of these techniques applied to the caches.

CHAPTER 3

PROCESS VARIATIONS IMPACT

3.1 Introduction

As technology scales, the feature size reduces, thereby requiring a sophisticated fabrication process. The variations increase as the feature size reduces due to the difficulty of fabricating small structures consistently across a die or a wafer [10].

Two main sources of variation are environmental and physical factors. The environmental factors depend on the operation of the system and include variations in temperature, power supply, and switching activity. The physical factors result from imperfections in the fabrication process and are permanent.

Physical factors can further be divided into random and systematic variations. Random variations are due to inherent unpredictability of the semiconductor fabrication process. Fluctuations in channel doping, gate oxide thickness, and ILD primitivity are primarily due to random variation. Systematic variation implies that changes in parameter values, such as L_{eff} , are due to known and predictable phenomena [17]. Systematic variations may be caused by a variety of different sources, most notably, variation in optical intensity across the exposure field and non-uniform chemical-mechanical polishing that occurs due to different pattern densities.

Process variations occur because specific steps in the fabrication process, such as lithography, ion implantation, and chemical-mechanical polishing, are vulnerable to imperfections, noise, and imperfect control across time and location [35].

The manufacturing process causes variations in many different parameters in the device, such as the effective channel length L_{eff} , the oxide thickness t_{ox} , and the threshold voltage

V_{th} . Controlling the variation in device parameters during fabrication is therefore becoming a great challenge for scaled technologies.

The performance of integrated circuits can be greatly affected by these variations because they can make a given circuit exhibit different delay or power characteristics than intended during design. The process variations are random in nature and are expected to become significant in the smaller geometry transistors commonly used in memories. Question is whether there is a significant enough delay variation overall that will drive a change in memory architecture design.

In this chapter, we analyze the impact of different sources of process variations in caches under worst-case and expected-behavior operating conditions. We are interested in exploring both performance and power consumption issues related to process variations and gathering insights that could be used in new cache designs as well as possibly in developing new architectural techniques for fetch units and load-store units in microprocessors.

In order to evaluate the impact of the parameter variations on circuit speed, we consider variability on the critical path. Indeed, the frequency (f), at which a circuit can be operated, is determined by the slowest path delay. The critical path of our CAM-tag cache is shown in Figure 3.1.

A CAM-based cache stores tags in a CAM array and stores data in SRAM arrays. CAMs perform tag checks in parallel. If a match is found, the cache provides the data in the associated cache line to the processor. Otherwise, a cache line replacement takes place.

The CAM-tag critical path in our design is composed of the global address decoder to select a bank, tristate I/O to drive the search bitlines, the dynamic match comparators in the CAM cells, wordline gating, the data SRAM array, column multiplexer, sense amplifier, and then the tristate I/O drivers connecting back to the CPU. The tristate bus that connects one 32-bit subbank column back to the CPU 32-bit load data path has the same fan-in in all configurations.

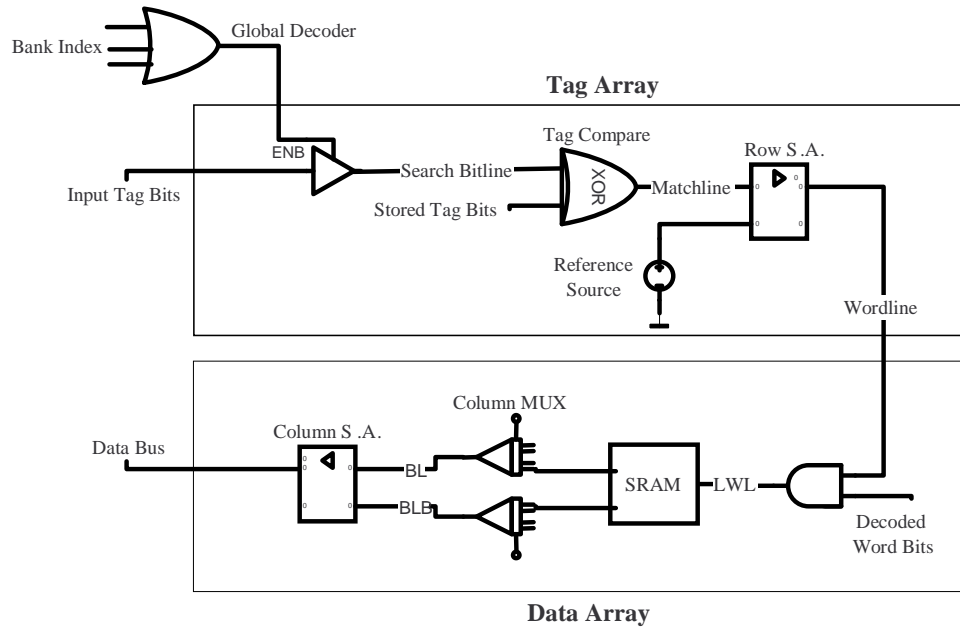


Figure 3.1. Critical path of CAM-tag cache.

Process variations in caches affect the performance of circuits such as sense amplifiers that require identical device characteristics and SRAM cells that require near-minimum-sized cell stability for large arrays in embedded, low-power applications. In addition, the delay of the address decoders suffers from the process variations that can result in shorter time left for accessing the SRAM cells.

In order to examine delay tradeoffs under process variations, we have evaluated the impact of process variations under both worst-case operating and typical behavioral conditions. The goal here is to establish a worst-case baseline that might be used in conventional conservative designs and a typical behavior that could be used to estimate the benefits of migrating to an adaptive design.

3.2 Worst-case Operating Conditions

Under worst-case operating conditions, we assume that parameter variations happen at each transistor in the cache critical path. We have used the HSPICE circuit simulator [2] at

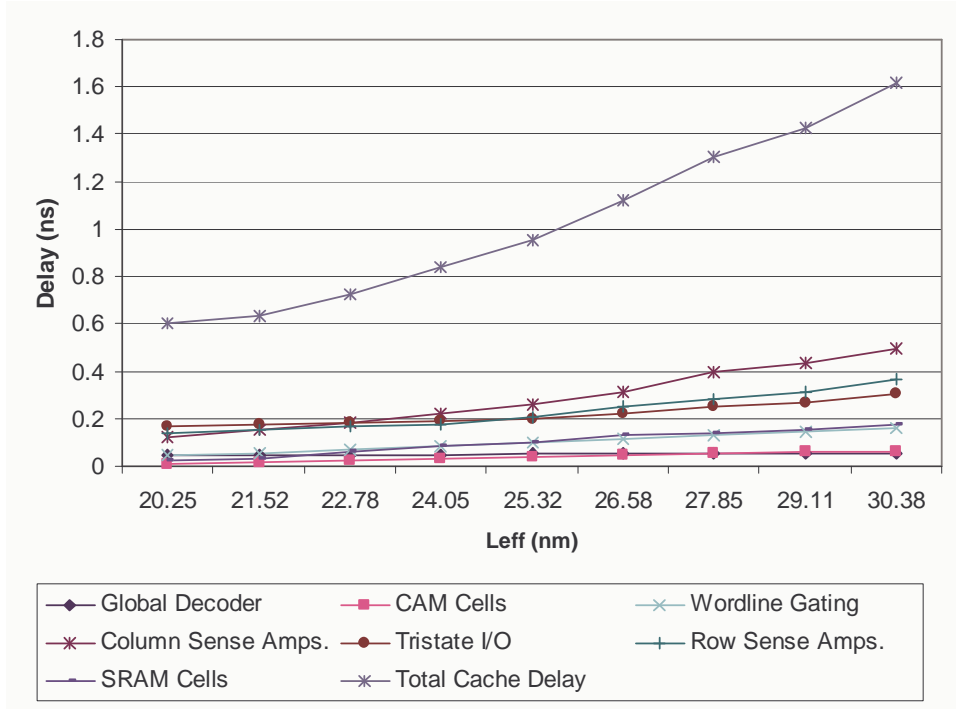


Figure 3.2. Effect of L_{eff} variation on cache delay.

the 32-nm PTM device model [1]. The nominal value used for V_{th} is 0.2V, and the nominal value for L_{eff} is 25.3nm, given by PTM technology.

3.2.1 Channel Length Variation

One of the most important sources of device variation is channel length variation (L_{eff}). There are numerous sources of channel length variation that affect the device at different stages throughout the fabrication process, such as etch rate variation, lithography dose variation, focus variation, lens imperfections, scanner variation, etc. Most of the gate length variation is due to imperfections in the lithographic system. These imperfections can contribute multiple levels of variation at any point during the fabrication process. For example, changes in the wafer machinery over long periods of time can introduce wafer-to-wafer variation. Also, within-wafer variation originates from non-uniformities in temperature, laser intensity fluctuations, or film-thickness changes [16].

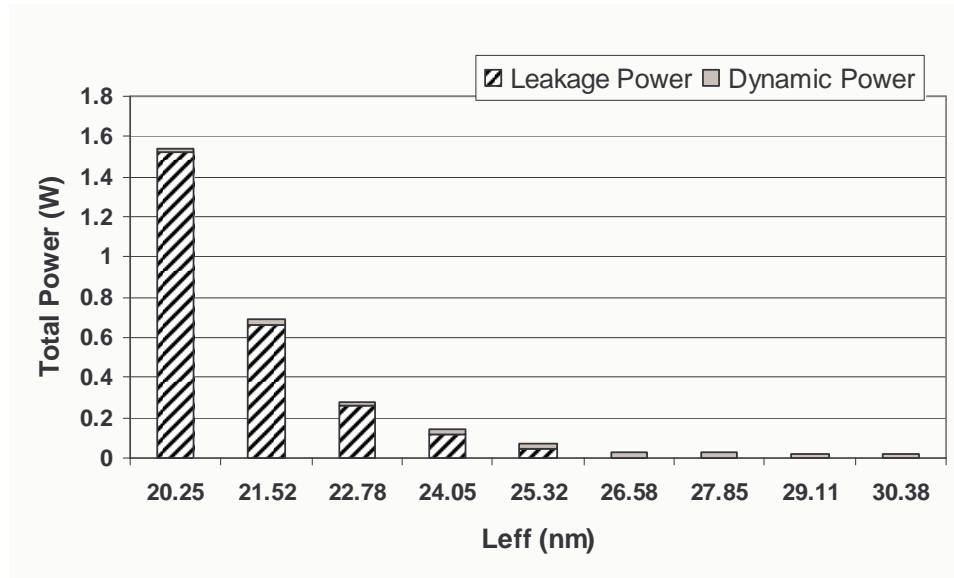


Figure 3.3. Effect of L_{eff} variation on power consumption.

Channel length variation (L_{eff}) is due to limitations in the lithographic process. Gate length variation can change the effective driving capability of the transistor, as well as the threshold voltage due to the short channel effect. These variations result in changes in device performance characteristics. A total of 40% variation in effective channel length (L_{eff}) is expected within a die [10]. We have found that the use of longer effective channel lengths tends to increase the wordline and bitline capacitances in caches, thus increasing access time, as shown in Figure 3.2. The access time can increase by as much as 2.7X.

Moreover, a small variation in the L_{eff} value causes a significant change in the power consumption for 1KB of SRAMs by as much as 60X (see Figure 3.3). The leakage power is dominant because we are reading a single word (32-bit) out of 1KB of SRAMs and no leakage reduction techniques are used here.

3.2.2 Threshold Voltage Variation

Threshold voltage can vary due to (1) changes in oxide thickness, (2) changes in the dopant levels in the substrate, polysilicon, and implants, and (3) surface charge. Accurate

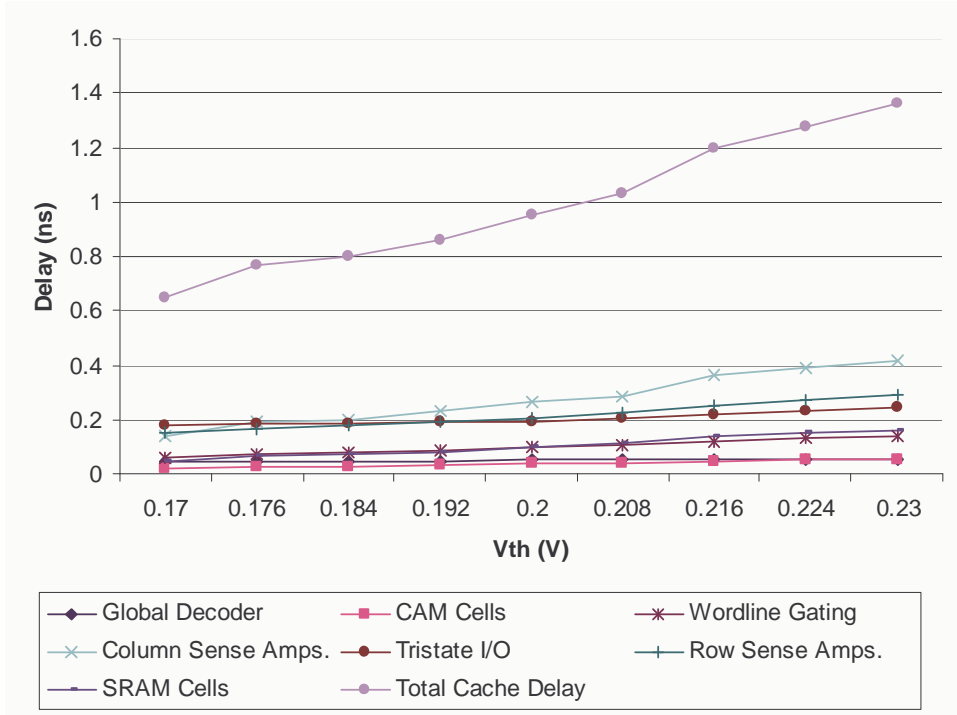


Figure 3.4. Effect of V_{th} variation on cache delay.

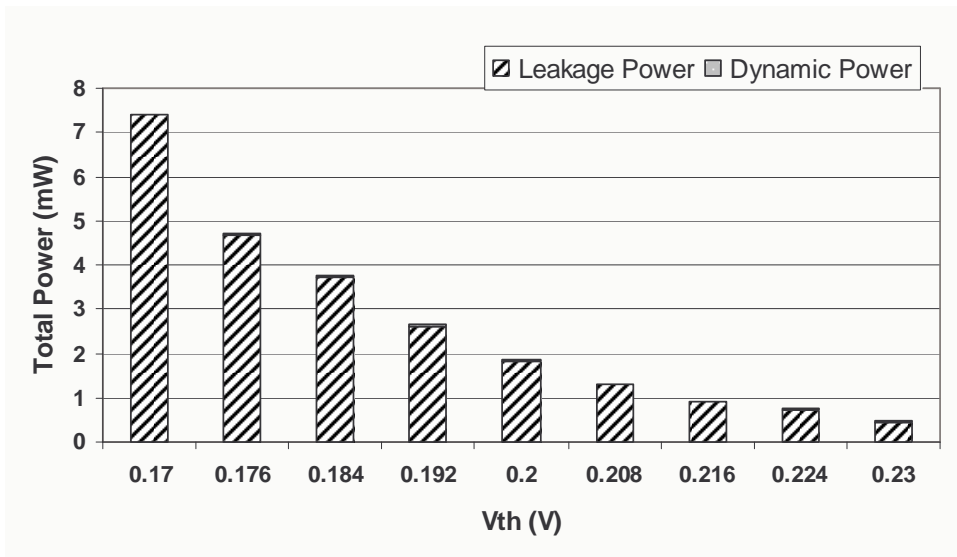


Figure 3.5. Effect of V_{th} variation on power consumption.

control of V_{th} is very important for many performance and power optimizations and for correct execution [60].

Higher transistor threshold voltage V_{th} , due to process variations, affects the access time due to the lower read current, as shown in Figure 3.4. The impact on the access time could be as much as 2.13X. As shown in Figure 3.5, the impact on leakage power could be as much as 17X.

3.2.3 Supply Voltage Variation

One of the most important environmental factors that cause variations in operating condition is supply voltage (V_{dd}). In deep submicron technology, the supply voltage is typically scaled down to reduce power consumption; effects such as the IR voltage drop and $L\frac{di}{dt}$ noise can affect the voltage level at the power supply, thereby modifying the characteristics of the transistors in the circuits. Moreover, power-saving mechanisms such as system standby and clock gating that control the switching activity of large circuit blocks within integrated circuits may also affect the power supply voltage level [49].

A total variation of 15% in V_{dd} was considered [10] with a nominal value of 0.9V. Table 3.1 summarizes our results showing delay and power consumption for our cache design. From Table 3.1, the power consumption of cache decreases significantly by 45% as V_{dd} level drops due to the quadratic relation between the dynamic power consumption and the power supply voltage. A reduction in supply voltage causes an increase in the access time of the cache by up to 12% of the nominal value.

Table 3.1. Effect of Power Supply Voltage Variations

V_{dd} (V)	Delay (ns)	Total Power (W)
0.83	0.746	0.183
0.86	0.717	0.187
0.90	0.667	0.191
0.93	0.634	0.213
0.97	0.601	0.266

The deviations in effective channel length and threshold voltage are shown to have a more significant contribution to the delay than variations in power supply voltage. There are many other process parameters that can have an impact on both power consumption and access time. The impact on cache access time due to process variations and longer wordline/bitline could become very significant. The worst-case access delay could be impacted by more than 3X when counting all the possible process parameters.

3.3 Expected Behavioral Operating Conditions

The simple use of worst-case values for all parameters that have been shown in Section 3.2 can result in larger path delay estimates than typical. These will certainly be pessimistic but would need to be considered in conventional designs. This section presents the delay distribution in a cache due process variation.

To accurately predict cache critical path delay distribution at the circuit level, cache delay variability can be studied through Monte Carlo in HSPICE circuit simulations. Process variations are typically represented by continuous probability distributions, and are often assumed to be normal distribution [13].

The distribution of delay of a cache critical path was determined by performing Monte Carlo sampling at different supply voltages, threshold voltages, and transistor lengths. Under the assumption of separated normal distributions of L_{eff} , V_{th} and V_{dd} variations, Monte Carlo simulations verify model predictions over a wide range of process and design conditions. We have used the Monte Carlo simulation with 5,000 trials where the variation sources all vary simultaneously. We simulate the critical path and measure delay with all the parameters varying with 3σ and mean values as specified in Table 3.2.

The probability density function (PDF) of the cache delay was measured (see Figure 3.6) for each process parameter. In addition, we have combined all the parameters in another experiment. We have found that most cache accesses under the impact of supply voltage or threshold voltage parameters would be relatively close to the nominal delay.

Table 3.2. Nominal and 3σ Variation Values for Each Source of Process Variations Modeled

Technology	32-nm	
Device	NMOS	PMOS
L_{eff}	25.32-nm ($\pm 20\%$)	
V_{th}	0.2V ($\pm 7.5\%$)	-0.2V ($\pm 7.5\%$)
V_{dd}	0.9V ($\pm 7.5\%$)	
Temperature	75°C	

Out of the three main process parameters discussed here, gate length variation remains the dominant source of delay variation. The deviations in L_{eff} are shown to have a significant contribution to the delay distribution (wider curve). It is also very close to the case with all the parameters combined.

Out of 5,000 random samples, assuming a 1-cycle cache at 1 GHz, 2,000 samples of the cache accesses are expected to be faulty, resulting in a probability of failure of 40%. However, with an increased cache delay of two cycles allowed and after adjusting the path across the components to accommodate a larger variation in the SRAM access, the probability that this cache will have to take two cycles has been found to be only 25%. This means that in an adaptive scenario only 25% of the accesses would require two cycles. We have also found that even in this case a small fraction of accesses would fail, suggesting that there are cases that would need three cycles for correctness.

3.4 Chapter Summary

Process variations will become worse with technology scaling; techniques are necessary at the architecture and circuit levels to reduce the impact of these variations while providing the highest performance for given power constraints. In this chapter, we have found significant delay variation between worst-case and expected behavioral analysis, motivating us to design an adaptive cache architecture. We have shown that process variation can have

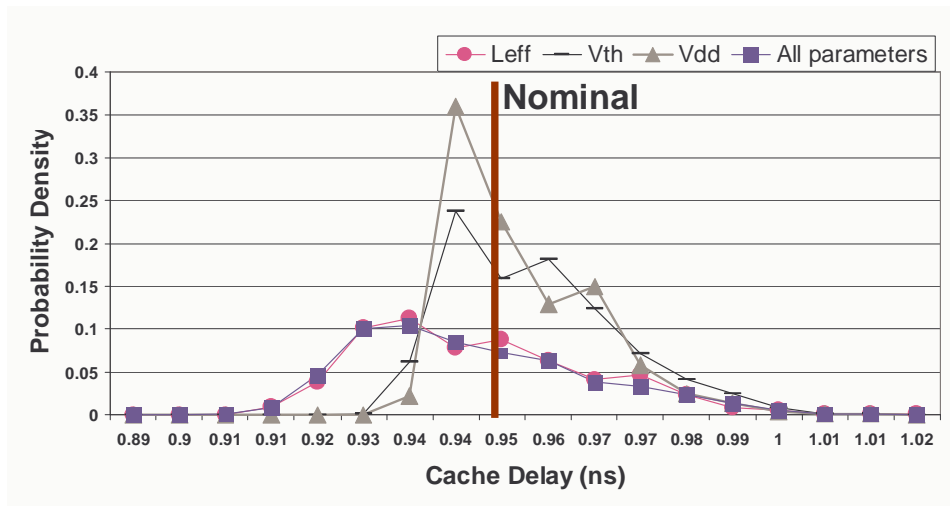


Figure 3.6. Distribution of the cache critical path delay.

a significant impact on delay (2-3X) under worst-case operating conditions, while under the expected condition a large fraction of accesses would be still close to the nominal value.

CHAPTER 4

RESILIENT DATA MEMORY DESIGN

4.1 Introduction

At the architectural level, we might be able to help to mitigate the negative impact of process variation such that low-power circuits and correct operations can still be applied. There are several ideas that could be exploited to cope with this problem while not giving up performance. These could range from using smaller first-level caches (that would meet the preferred access time even under worst-case variation) to more adaptive cache architectures that we will present next.

4.2 Conservative Cache

As we have shown in the previous chapter, process variations affect the latency significantly for each cache access. The cache access latency difference could be as much as 3X if we consider all the possible variations in process parameters. The conservative cache would have to be based on the worst-case process variation analysis (as shown in Section 3.2). Alternatively, one could make the cache access time slightly more aggressive (than the conservative one), but then the yield would be likely affected.

In a conservative cache design, to ensure the correct execution in the pipeline architecture, the cache access delay cycles must be decided based on the longest delay possible within the process variation range. For example, even a small latency increase due to variation may require the whole cache access latency to be increased by one or more processor cycles. This can severely degrade the overall application performance.

For example, even if we could access 90% of the cache lines within one cycle, for the remaining 10%, we might need two or three cycles to finish the access due to the delay increase resulting from process variations. In this situation, we might need to assume the worst-case scenario, which is 3 cycles for all the cache accesses. This will, as a result, severely affect the total performance and is clearly not a choice that we can use even if some architectural tricks could be applied to hide the cost (e.g., overlapping memory access with other instructions).

For example, clever scheduling techniques might try to increase the distance between memory reads and consumer instructions to hide a longer latency. As we have seen, however, there is a limit to how much that can help as very often basic blocks are short and the scheduler cannot move dependent instructions several cycles away.

4.3 Proposed Adaptive Cache

In the proposed adaptive cache design, instead of accessing the cache with a long fixed latency assuming worst-case conditions, the adaptive architecture can have different access latency for different cache lines. The typical case analysis encourages efforts toward developing adaptive design methodologies that suppress the impact of process fluctuations on performance. The expectation is that most of the cache lines will have much lower latency compared to the worst-case scenario.

Figure 4.1 shows a possible adaptive architecture. One of the important blocks in the proposed architecture is the delay storage unit. This unit stores the speed information and is read along with the data array on every cache access. The operation on the delay storage has two phases: classification and execution.

Delay information for each cache line is first achieved during a classification process in which each cache line is probed individually and its delay information is written into the delay storage unit. Then, during the execution phase, the delay information is fetched from the delay storage, and each cache line can be accessed based on its estimated speed.

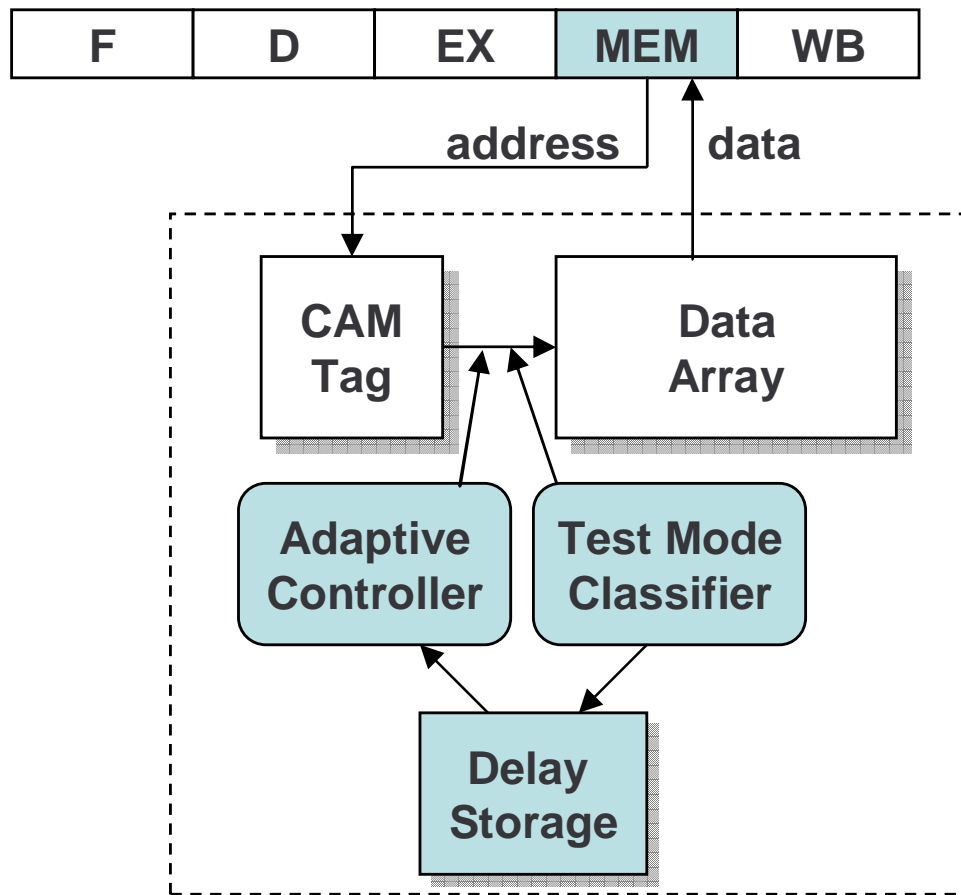


Figure 4.1. The proposed adaptive cache architecture (shown in a single five-stage pipeline).

With the addition of the delay storage, we are able to access the cache with an adaptive speed. The adaptive architecture will enable us to maximize the performance compared to the traditional fixed latency cache architecture.

4.4 Mechanism and Implementation

Figure 4.2 shows the proposed delay-resilient cache architecture during the classification phase. The cache is equipped with a BIST circuitry, which tests the entire cache and detects the speed of each cache line during the classification phase.

The basic idea is let the sense amplifiers to sample the bitlines during the read operation in three stages. In the first stage, the sense amplifier will sense the bitlines in one clock cycle. While in the second stage, the sense amplifier will take two cycles to sense the bitlines, and in the third stage, the sense amplifier has to be fired in three cycles.

During the classification phase, if a cache component is affected by process variations (for example, higher V_{th} in SRAM cell), the SRAM cell might not be able to establish enough voltage differentials between bitlines by the end of the first cycle; thus the access delay is more than one cycle.

In the second stage, the sense amplifier is fired in two cycles. If the sense amplifier still cannot get the value that has been previously stored in the SRAM cell, the delay is more than two cycles. In the third stage, the sense amplifier will be fired in three cycles. Therefore, comparing the outputs of each sense amplifier with the original value stored, the actual cache access latency for this cache line can be identified. With this technique, a maximum delay of three cycles can be detected (a failure will occur if the actual latency is more than three cycles). This delay information will be stored in the delay storage unit.

Each cache line is tested using BIST when the test mode signal is on. For example, a block is considered fast, medium, or slow. BIST feeds this information into the delay storage.

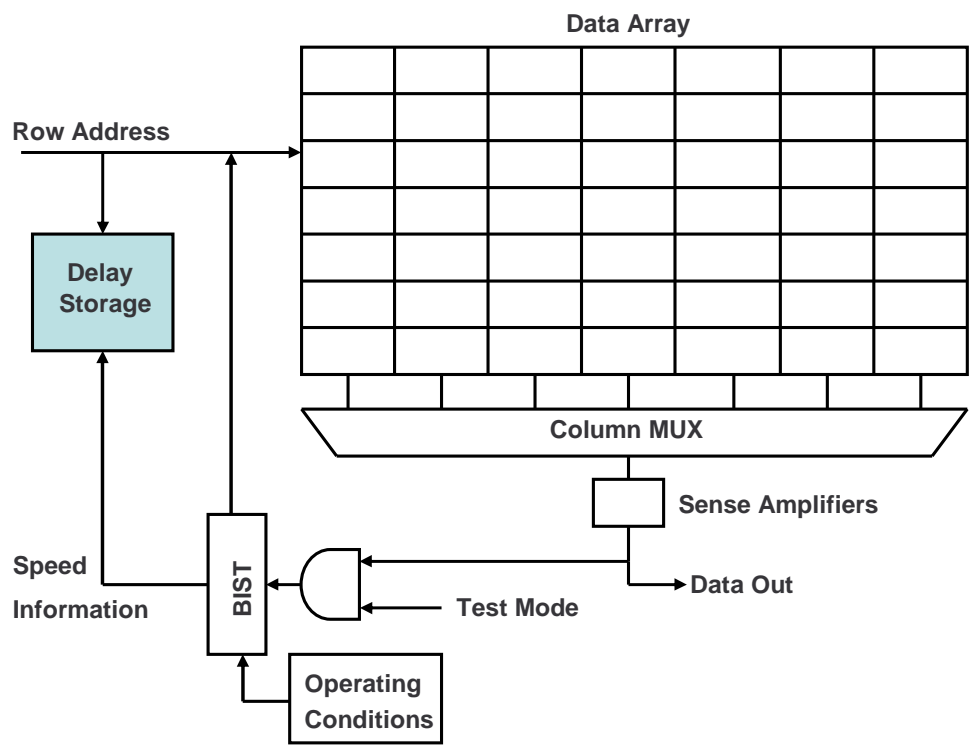


Figure 4.2. The adaptive cache architecture during the classification phase.

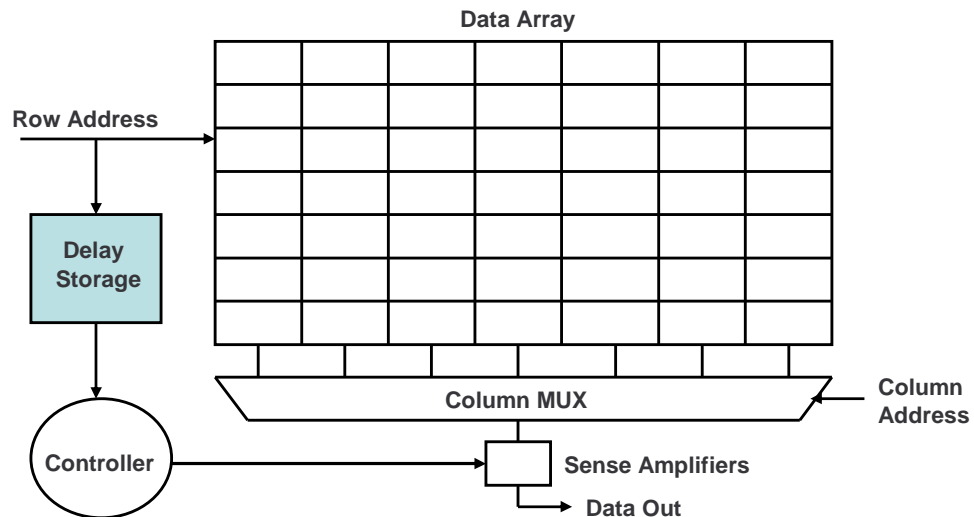


Figure 4.3. The adaptive cache architecture during the execution phase.

The delay storage is implemented as a small memory array of two bits per cache line. Therefore, each row in the delay storage stores the speed information of all the SRAM cells in each cache line. For example, the two bits are “11” if the corresponding cache line latency is three cycles, “10” if the corresponding latency is two cycles, or “01” for one cycle (“00” can be also used to indicated a failure). These bits are determined at the time of testing and stored by the BIST circuit.

Delay storage provides the speed information to the controller. The speed information stored in the delay storage is used by the controller to control the speed of the sense amplifiers during regular read operations. The controller can be built by using combinational circuits that will provide the enable signals to the sense amplifiers at different speed.

During the execution phase (Figure 4.3), both the data array and delay storage units are accessed in parallel using the row address of the index bits. The delay information is fetched from the delay storage and fed to the controller. If the cache line is slow, the controller will delay the enable signals to the sense amplifiers. The sense amplifiers would need to be triggered at different time points depending on the speed access.

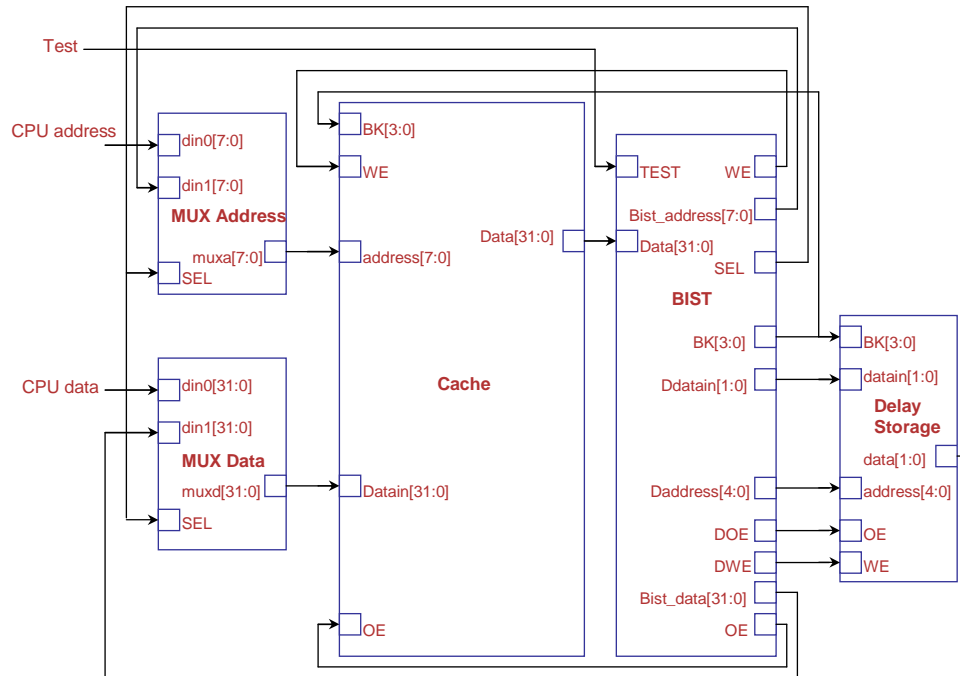


Figure 4.4. BIST circuit.

This scheme is transparent to other subsystem and has negligible power and area overhead. The area penalty for this cache is minimal, as we only need to use two bits (our example with one to three cycles latencies) for each cache line (or 256 bits) to encode the speed.

4.5 Results and Analysis

In the previous chapters, we presented the power and the delay as well as the process-variation-related implications in our low-power cache. This chapter provides an analysis of the new added components to our adaptive cache. In our analysis, we have evaluated the area overhead associated with the extra BIST, delay storage, and control circuitry by using the Synopsys Design Compiler tool to generate the netlist for the extra hardware needed for the adaptive cache (see Figure 4.4).

We have found the overall area overhead to be less than 1% of the total cache area (see Table 4.1). Because the delay storage is a small structure (two bits per cache line), and not on the cache critical path, its own delay variation due to process variation is small compared to the cache. Since the data array and delay storage units are accessed in parallel during the execution phase, there will be no delay overhead by introducing the delay storage. In addition, we have evaluated the power consumption overhead associated with this extra circuitry; we have found the overhead to be 2% of the total cache power for a single word read.

Table 4.1. Overhead Associate with the Adaptive Design

Circuit	BIST, delay storage, and control circuitry	Cache
Delay	No delay overhead	0.95 ns
Power	0.55 mW	27.67 mW
Area	0.0048 mm^2	0.54 mm^2

4.5.1 Performance Speedup

The adaptive cache architecture is implemented in the SimpleScalar architecture simulator [14]. Simulation parameters are summarized in Table 5.7. We have conducted simulations of SPEC2000 benchmarks [3] using the adaptive approach. We varied the cache access latency from one to three cycles. The adaptive cache based on the delay distribution is determined by the Monte Carlo simulation. Based on our analysis, the adaptive cache is expected to have 75% of one-cycle, 24% of two-cycle, and 1% of three-cycle cache line accesses.

Results for the application performance are shown in Figure 4.5. A comparison is made between a conservative cache that requires three cycles per access and an adaptive cache that has variable cache access latencies.

Our results show that the adaptive cache design in a four-way issue processor can achieve a 9% to 21% (on average, 17%) performance improvement for the applications

Table 4.2. SimpleScalar Parameters for CPU

Parameter	Value
IQ/LSQ/RUU	4/8/16
Fetch, dispatch, commit width	4
Issue Width	4, out-of-order
Integer ALU/mult-div	4/1
FP ALU/mult-div	4/1
L1 D-cache Size	16KB, 32-way associative, 32-byte per block (LRU)
L1 I-cache Size	16KB, 32-way associative, 32-byte per block (LRU)
L2 Unified Cache Size	128KB, 64-way associative, 64-byte per block, 8 cycles (LRU)
Memory Bus Width/latency	8 bytes/100 cycles
Memory Ports	2
I-TLB Size	64-entry, fully associative, 30-cycle miss penalty
D-TLB Size	128-entry, fully associative, 30-cycle miss penalty
Branch Predictor	Combination of bimodal and 2-level gshare; bimodal size 2048; level1 1024 entries, history 10; level2 4096 entries (global)
Branch Target Buffer	512-entry, 4-way associative
Return-address-stack	8-entry

studied, while providing resilience against failures due to the process variation comparison made to a conservative design assuming worst-case latency.

4.5.2 Sensitivity to Issue Width

Figure 4.6 shows the performance speedup improvement for different instruction issue widths on several SPEC2000 applications. Speedup values are normalized with respect to the worst-case delay of three cycles. As we can see, the eight-way issues design benefits more than the four-way issues from the adaptive cache architecture.

Using the adaptive cache architecture can also mean that one can set the clock rate slightly more aggressively: the increase in clock rate would likely compensate for a larger

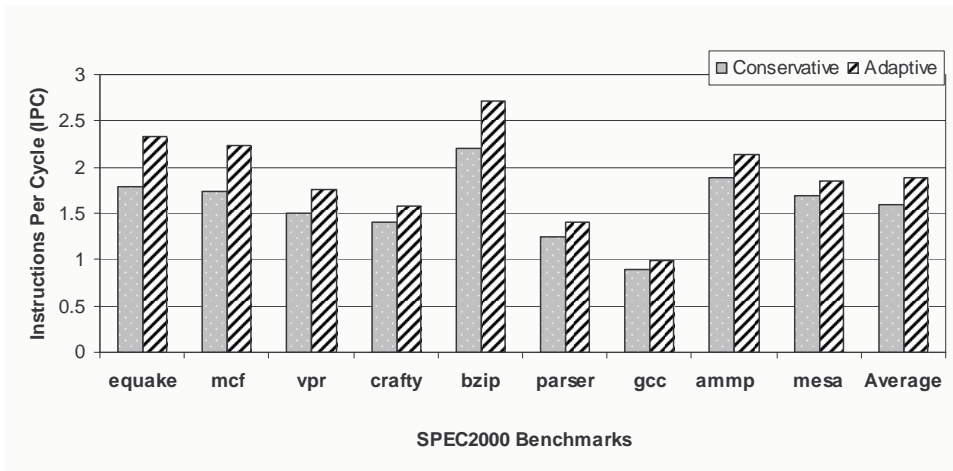


Figure 4.5. Performance improvement between adaptive cache vs. a conservative cache using three-cycle access time.

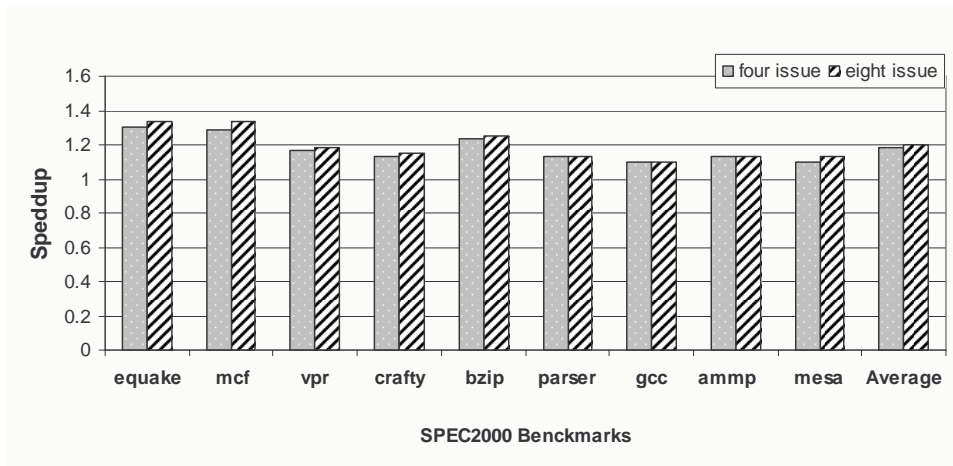


Figure 4.6. Comparison of performance speedup for 16KB cache on four-way issue and eight-way issue SimpleScalar architecture.

fraction of memory accesses falling into higher-latency memory access categories in a processor. Furthermore, when low power is important, a slightly slower cache (e.g., due to leakage enhancement cell (LE) designs [8] with some high- V_{th} transistors to reduce cell power) would mean a redistribution between one-, two-, and three-cycle accesses.

Non-uniform cache architecture (NUCA) has been proposed as a solution to the problem of growing wire delay and clock rates that limit the cache access within a single cycle [39]. In NUCA, the cache is physically portioned into banks that can be accessed at different latencies. NUCA allows fast access to close banks while retaining slow access to far banks. Frequently accessed data are placed in banks closer to the processor while infrequently accessed data are placed farther away.

In contrast, in our cache we can achieve different cache access latency within the same cache bank. To account for process variations, the cache is logically portioned into lines, each with different access latencies. Our objective is different; we propose an adaptive variable-cycle-latency cache architecture that mitigates the impact of process variations on access latency. Our adaptive cache design can, however, be applied to each NUCA bank to mitigate the impact of process variations.

4.6 Chapter Summary

Process variations will become worse with technology scaling; techniques are necessary at the architecture and circuit levels to reduce the impact of these variations while providing the highest performance for the given power constraints. The proposed adaptive cache architecture can improve the application performance in a superscalar design by as much as 21% depending on the application and configurations used, compared to a conservative design. This scheme is transparent to other subsystems and has negligible power and area overhead. The adaptive cache architecture also allows a designer to choose the first-level cache access latency more aggressively and possibly increase the clock rate in a processor design where cache access is the main critical path. Furthermore, The proposed cache

architecture could help strike a better balance between power and delay optimizations in a design.

CHAPTER 5

POWER AND PERFORMANCE TRADEOFFS WITH PROCESS VARIATION RESILIENT ADAPTIVE CACHE ARCHITECTURES

5.1 Introduction

The power consumption and performance of caches can be greatly affected by process variations and therefore a major challenge for processor designers who need to meet power budgets and achieve desired performance levels consistently. It is clear that to address this challenge we will likely need a combination of solutions at different layers, such as circuits and architecture. We need to ensure that circuits and architectures are resilient to delay variations, without giving up performance or consuming more power than nominally budgeted.

In Chapter 4, we have proposed an adaptive cache architecture that mitigates the impact of process variations on cache delay. Our objectives in this chapter are (1) to analyze the effects of various parameter variations on the overall cache leakage power before and after applying leakage reduction optimizations; (2) to observe how leakage reduction techniques are affected by the variation: we investigate applications of leakage enhancement (LE) SRAM cells [8] to reduce leakage power; (3) what the expected leakage power distribution is in a complete cache; and (4) how an adaptive cache architecture would be able to control the tradeoffs between leakage and performance, including capturing effects at both circuit and application levels in a superscalar microprocessor.

We show that one can minimize power consumption while meeting the operating frequency constraints under the impact of process variations and with minimal effect on application performance compared to the optimal performance setting in the adaptive

cache. Furthermore, we show that compared to a conservative design, an adaptive cache architecture resilient to process variations can even achieve an improvement on application performance in conjunction with a 6X lower leakage despite the effects of process variation.

5.2 Impact of Process Variation in Caches

5.2.1 Cache Leakage Power

In current CMOS technologies, leakage current is composed of three major sources: subthreshold leakage, gate leakage, and substrate leakage. Subthreshold leakage is the main source of leakage in current and future technologies, especially now that the accelerated adoption of high-k gate dielectric is set to reduce gate leakage [21]. Subthreshold leakage is the current that flows from drain to source even when the transistor is off.

A six-transistor (6T) SRAM cell serves as the basic construction block of a standard cache design. The basis of a 6T cell is a pair of cross-coupled inverters with two access transistors (see Figure 5.1). The majority of the SRAM cells spend their time in the standby state. In this state, most of the leakage power is dissipated by the transistors that are off and have a voltage differential across their drain and source. As shown in Figure 5.1, there are three leakage paths in one 6T SRAM cell since only one transistor is “off” along each path. Such a SRAM structure consumes considerable amounts of static current in order to preserve data in the cells.

To make matters worse, process variation magnifies the leakage problem. Since there is an exponential relationship between leakage current and threshold voltage, increasing variations in threshold voltage can cause a severe variation in leakage power across the cache.

In this section, we also analyze the effects of parameter variations on our adaptive cache architecture. We target this design for a future 32-nm technology process where leakage and process variation will play prominent roles.

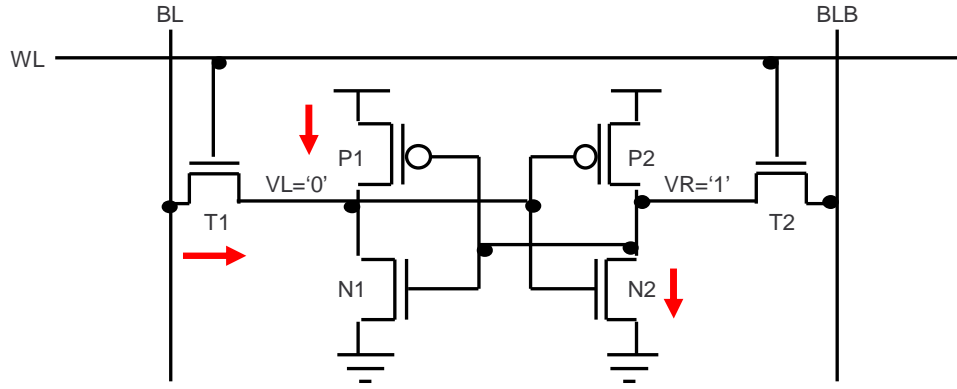


Figure 5.1. The structure of a basic 6T SRAM cell.

To accurately predict leakage power distribution at the circuit level, cache power variability can be studied through Monte Carlo in HSPICE circuit simulations. We simulate the 16KB cache and measure leakage power with all the parameters varying with 3σ and mean values as specified in Table 5.1. These assumptions are comparable to the data forecast in [10]. We run the simulation for 10,000 random samples. Process variations are typically represented by continuous probability distributions, and are often assumed as normal distribution [13].

Table 5.1. Nominal and 3σ Variation Values for Each Source of Process Variations Modeled

Technology	32-nm	
Device	NMOS	PMOS
L_{eff}	25.32-nm ($\pm 20\%$)	
V_{th}	0.2V ($\pm 7.5\%$)	-0.2V ($\pm 7.5\%$)
V_{dd}	0.9V ($\pm 7.5\%$)	
Temperature	75°C	

5.2.2 Impact of Channel Length Variation on Leakage Power

Channel length variation (L_{eff}) is due to limitation in the lithographic process. Channel length variation can change the effective driving capability of the transistor, as well as the threshold voltage due to the short channel effect.

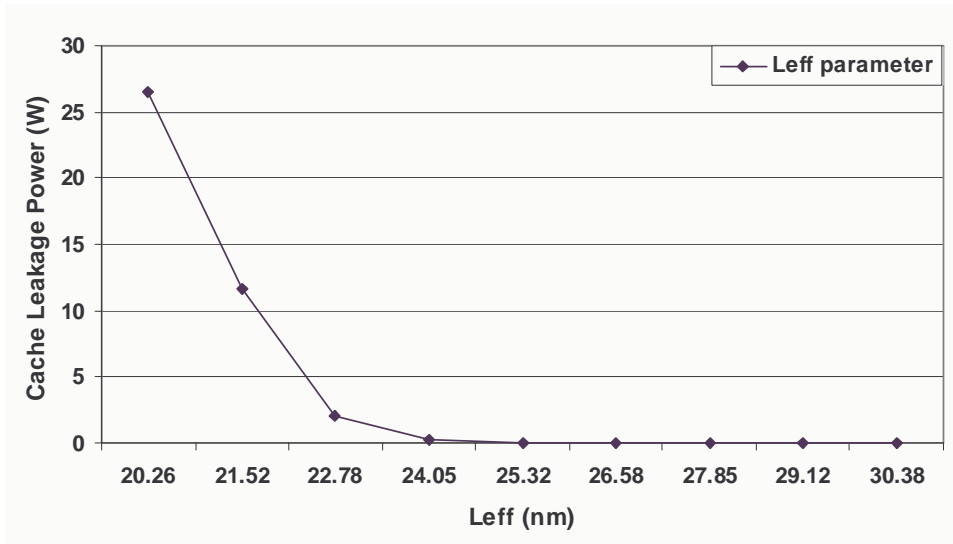


Figure 5.2. Cache leakage power as a function of L_{eff} .

Figure 5.2 shows the cache leakage as a function of L_{eff} . Leakage is exponentially dependant on channel length. The nominal value for the cache leakage when there is no process variation is 0.032W. Note that a 10% variation in a transistor's effective channel length causes as much as 60X from the nominal value!

Figure 5.3 shows the probability distribution of the cache leakage power for different percentage of L_{eff} variations. The figure and Table 5.2 show that, as L_{eff} variation increases, the mean leakage power of the cache increases, and the leakage power distribution gets more spread out. For a total 40% percent variation in the effective channel length of the transistor, there can be up to a 14X difference in the amount of mean leakage power compared to the nominal value (refer to row 4 at column 5 of Table 5.2).

Table 5.2. Effect of Channel Length Variations

Percentage of variations in L_{eff}	10%	20%	30%	40%
Maximum leakage power (W)	0.252	2.890	11.67	25.27
Minimum leakage power (W)	0.010	0.003	0.002	0.001
Mean leakage power (W)	0.038	0.060	0.181	0.462
Sigma (σ)	0.017	0.109	0.511	1.365

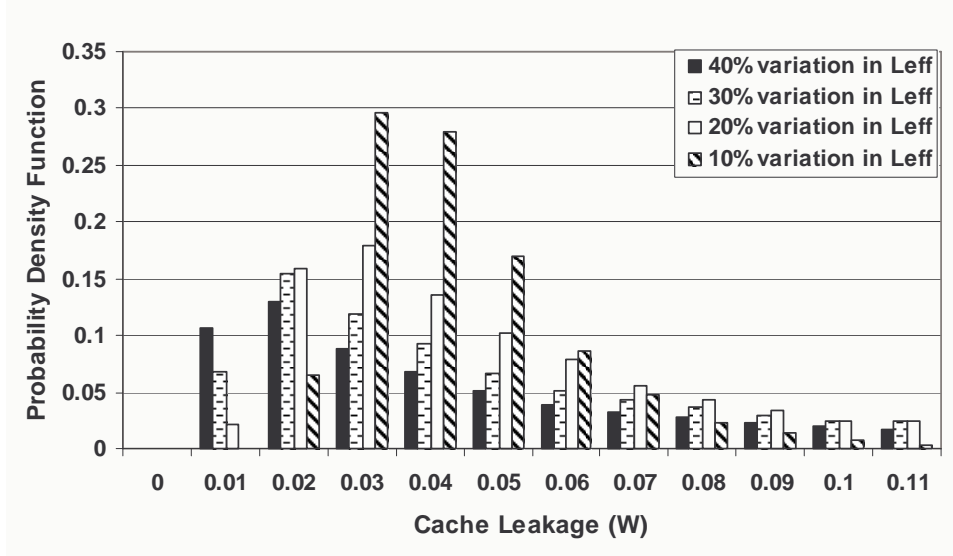


Figure 5.3. Probability distribution of the cache leakage power as a function of L_{eff} .

5.2.3 Impact of Threshold Voltage Variation on Leakage Power

A key process parameter subject to variation is the transistor threshold voltage (V_{th}). Figure 5.4 plots the cache leakage power as a function of V_{th} . Leakage power increases exponentially with decreasing threshold voltage. We found that small variations in device threshold voltage result in leakage numbers that differ by a factor of 2.

Figure 5.5 shows the probability distribution of the cache leakage power for different percentages of V_{th} variations. As we increase the percentage of variation in V_{th} , the maximum leakage power of the cache increases, and the leakage power distribution gets more spread out. Table 5.3 summarizes our results that show leakage power for our cache design.

Table 5.3. Effect of Threshold Voltage Variations

Percentage of variations in V_{th}	5%	10%	15%
Maximum leakage power (W)	0.0392	0.0484	0.0592
Minimum leakage power (W)	0.0251	0.0196	0.0176
Mean leakage power (W)	0.0322	0.0325	0.0330
Sigma (σ)	0.0018	0.0037	0.0056

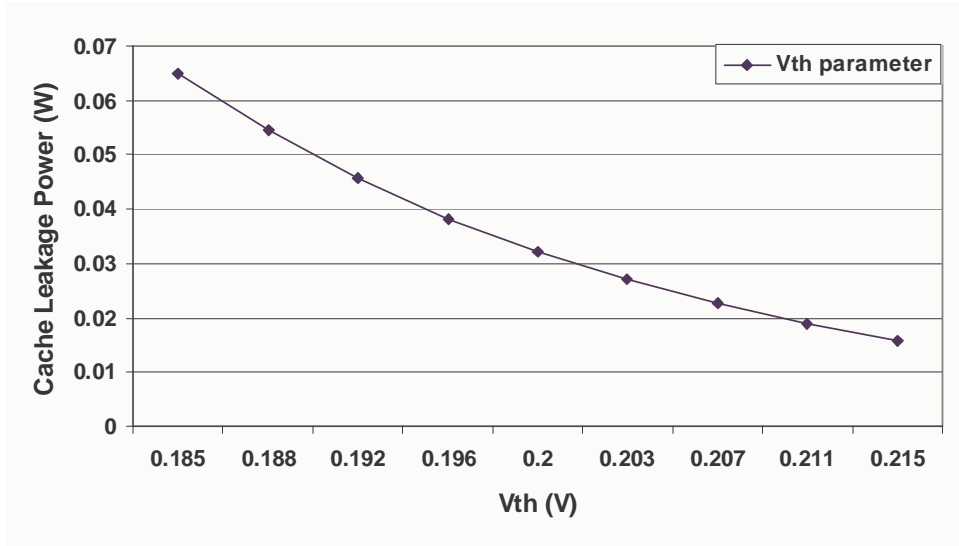


Figure 5.4. Cache leakage power as a function of V_{th} .

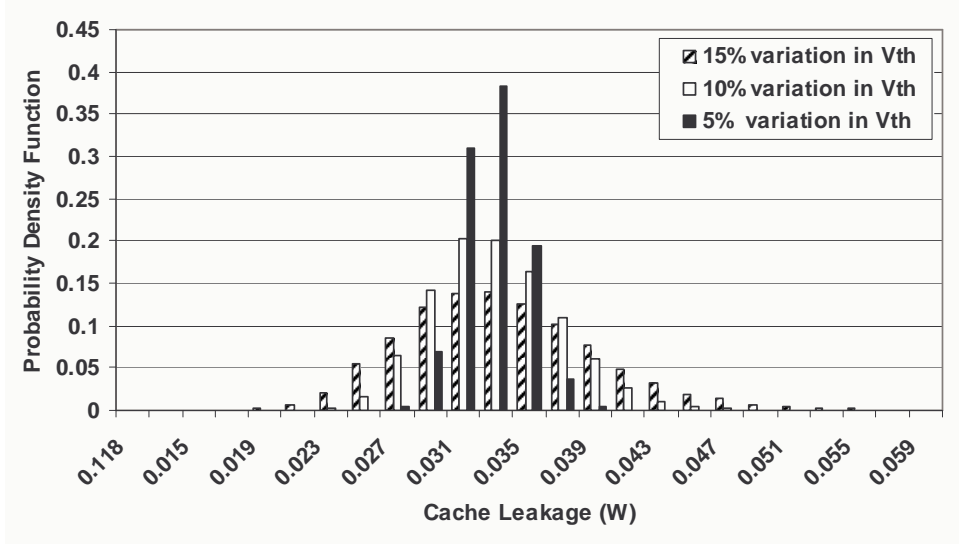


Figure 5.5. Effect of V_{th} variation on cache leakage.

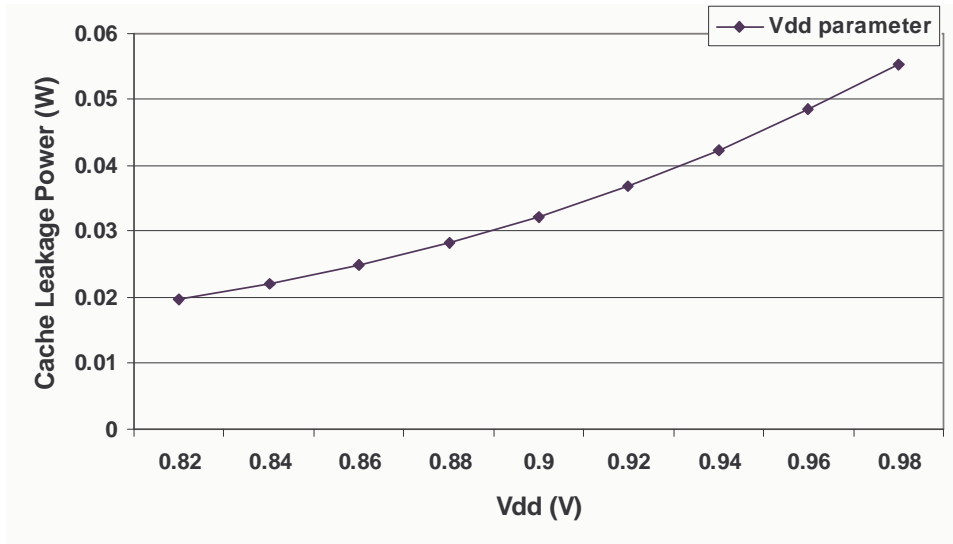


Figure 5.6. Cache leakage power as a function of V_{dd} .

5.2.4 Impact of Supply Voltage Variation on Leakage Power

One of the most important environmental factors that cause variations in operating conditions is supply voltage (V_{dd}). Figure 5.6 plots the cache leakage power as a function of V_{dd} with a nominal value of 0.9V. In the figure, a reduction in supply voltage causes a decrease in the leakage power of the cache by up to 1.7X of the nominal value. Thus, the supply voltage has a significant impact on the leakage power of the cache.

Figure 5.7 shows the probability distribution of the cache leakage power for a total variation of 5%, 10%, and 15% in V_{dd} . For a 15% percent variation in the supply voltage parameter, there can be up to a 2X difference in the amount of maximum leakage power compared to the nominal value (refer to row 2 at column 4 of Table 5.4).

Table 5.4. Effect of Power Supply Voltage Variations

Percentage of variations in V_{dd}	5%	10%	15%
Maximum leakage power (W)	0.0408	0.0518	0.0629
Minimum leakage power (W)	0.0268	0.0221	0.0204
Mean leakage power (W)	0.0322	0.0324	0.0326
Sigma (σ)	0.0016	0.0033	0.0046

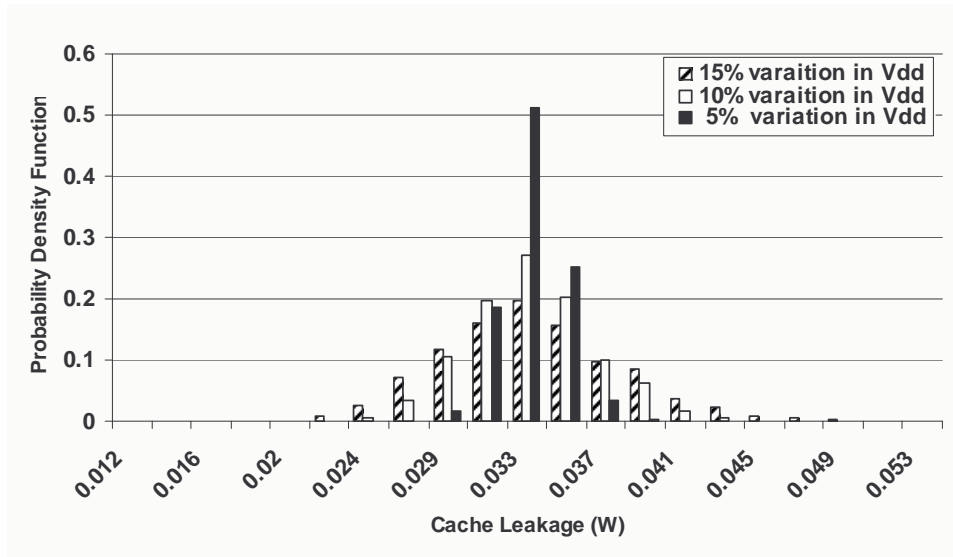


Figure 5.7. Probability distribution of the cache leakage for different values of V_{dd} .

5.2.5 Impact of All Parameters Combined

As we have shown in the previous section, the deviations in effective channel length have a more significant contribution to the cache leakage power than variations in threshold voltage and power supply voltage.

The probability density function (PDF) of the cache leakage power was measured when the variation sources all vary simultaneously (see Figure 5.8). The cache leakage distribution is very close to the case with the L_{eff} parameter variations.

We have found that the impact on cache leakage power due to process variations could become very significant. The mean leakage power could be impacted by 15X (compared to the nominal value) when counting all the possible process parameters (see Table 5.5).

This is a very serious variation as designers would have a difficult time meeting power budgets under these constraints. But before proposing a new adaptive technique to mitigate the effects, let us review how the leakage power varies when we apply leakage optimizations. Are these effects equally serious in designs with state-of-the-art leakage optimizations?

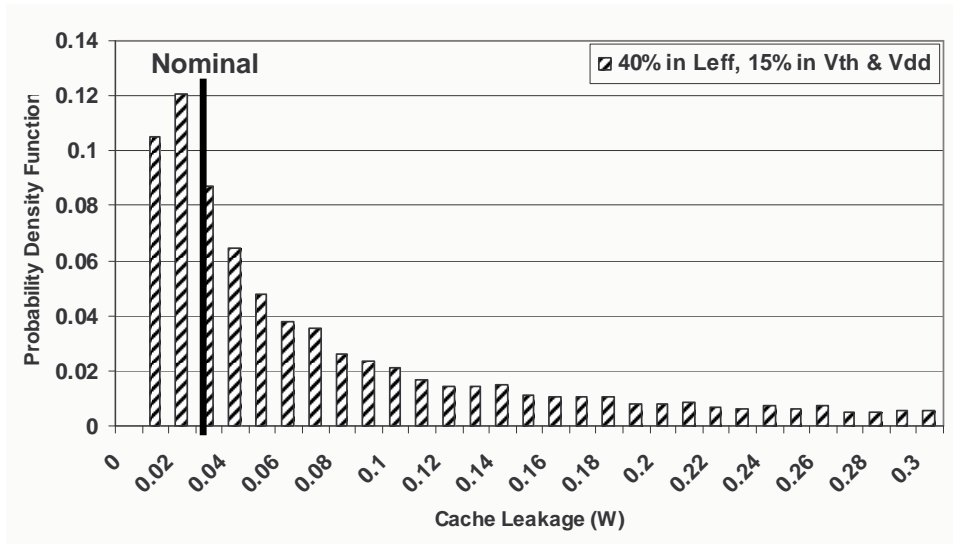


Figure 5.8. Distribution of the cache leakage power.

Table 5.5. Effect of Parameter Variations

Percentage of variations	40% in L_{eff} , 15% in V_{dd} and 15% in V_{th}
Maximum leakage power (W)	32.19
Minimum leakage power (W)	0.0013
Mean leakage power (W)	0.467
Sigma (σ)	1.32

5.3 Leakage Reduction Technique

Several research and industrial groups have introduced circuit and architectural techniques to reduce leakage power. One useful technique for reducing leakage power at the circuit level is the leakage enhancement (LE) SRAM cell [8].

Figure 5.9 shows a leakage enhancement SRAM cell. In the inactive state, when the cell is not being written to or read from, most of the leakage power is dissipated by the transistors that are off and that have a voltage differential across their drain and source. If the cell were storing a “0,” transistors T1, N1, and P2 would dissipate leakage power. A simple technique for reducing leakage power would be to replace all transistors with high- V_{th} ones, but this degrades the bitlines discharge times, affecting cell read performance

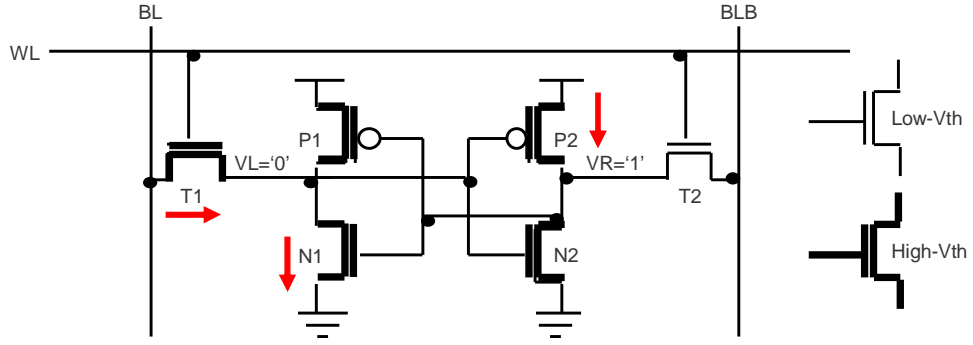


Figure 5.9. Leakage of enhanced SRAM cell [8].

significantly. In our design, we instead applied the same high- V_{th} for all the NMOS transistors (even transistor T2 in Figure 5.9).

In the following, we examine the power/performance tradeoffs of this LE technique applied to the adaptive cache under the impact of the process variations. To analyze the effectiveness of the leakage enhancement cell scheme, we need to analyze two different aspects: its impact on power and performance. The leakage power for different high- V_{th} values is measured, each with a different delay cost. We assume a low- V_{th} to be 0.2V. The optimal high- V_{th} is therefore the one that results in the minimum leakage power consumption while meeting the target frequency.

Figure 5.10 shows cache leakage power distribution using an LE cell with different high- V_{th} values. By increasing the transistor threshold voltages (high- V_{th}), the leakage power distribution moved toward the nominal value.

Table 5.6 shows that as we increase high- V_{th} , the leakage power decreases. In addition, applying 0.3V for high- V_{th} reduces cache average leakage by more than 6X compared to the nominal threshold voltage and decreases the delay by 16%.

The threshold voltage of the transistor can be controlled by using process adjustment (increasing channel doping to increase V_{th}) or body bias (BB). A body bias is a voltage applied between the source or drain of a transistor and its substrate, effectively changing the transistor's V_{th} [61].

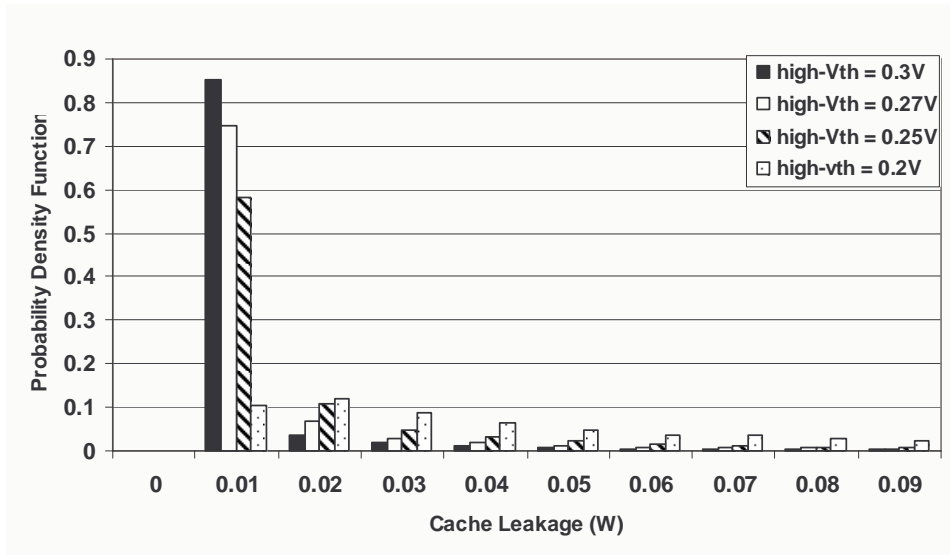


Figure 5.10. Cache leakage power distribution using leakage enhancement cells.

Table 5.6. Effect of Parameter Variations Using Leakage Enhancement Cells

High- V_{dd}	0.2V	0.25V	0.27V	0.3V
Delay (ns)	0.952	1.091	1.091	1.122
Maximum leakage power (W)	2.19	29.58	28.51	27.44
Minimum leakage power (W)	0.0013	0.00017	0.00011	0.00003
Mean leakage power (W)	0.467	0.182	0.116	0.076
Sigma (σ)	1.32	0.97	0.82	0.71

Depending on the polarity of the voltage applied, V_{th} increases or decreases. If it increases, the transistor becomes less leaky and slower (reverse body bias); if it decreases, the transistor becomes leakier and faster (forward body bias). In reverse body bias, the threshold voltage increases by raising the voltage of the PMOS n-wells with respect to the supply voltage or by lowering the voltage of the substrate relative of the NMOS transistor to the ground.

The PMOS and NMOS body bias voltages that result in the high-threshold voltage (high- V_{th}) may be applied by an on-chip body bias generator. The body bias generator generates separate body bias voltage for PMOS and NMOS transistors. The variable

threshold voltage scheme is conceptually illustrated in Figure 5.11. Note that this scheme could achieve the static LE scheme by applying body biasing on the FETs in the SRAM cells. The BB scheme can be also controlled dynamically. In the adaptive architecture we describe in the following section, BB can be used to adapt the architecture for various power performance tradeoffs while keeping the design process variation resilient.

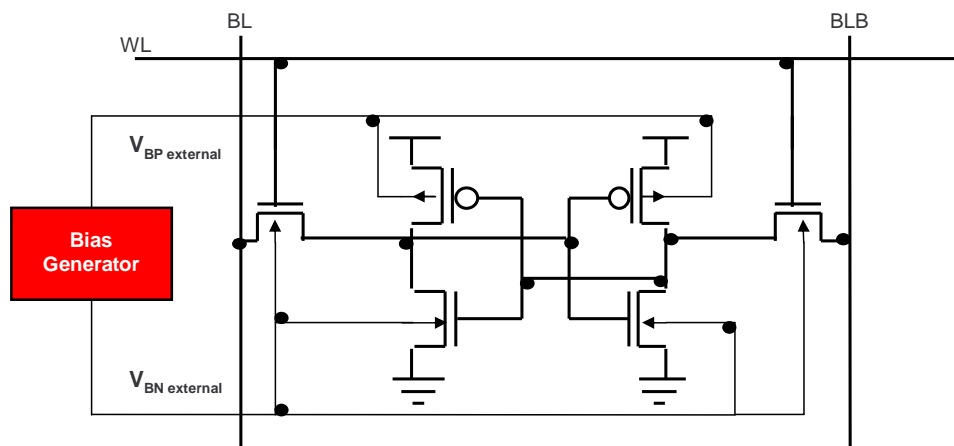


Figure 5.11. Variable threshold-voltage scheme.

5.4 Adaptive Cache Architecture

In Chapter 4, we have proposed an adaptive cache architecture that mitigates the impact of process variations on access latency. In essence, what the architecture does is it allows accesses to take variable cycle latency based on initial classification. The architecture works well because most of the accesses would still be close to the nominal latency and the of accesses on the tail of the delay distribution are relatively infrequent.

Our results have shown that the adaptive cache architecture can achieve a 9% to 21% (on average, 17%) performance improvement on the SPEC2000 applications studied compared to a conservative design based on worst-case latency (three cycles) due to process variation, while providing process variation resilience.

We extended our adaptive cache by using the BB; Figure 5.12 shows our proposed static adaptive architecture. We call it static adaptive cache because the optimum bias voltages are determined through measurements on our adaptive cache at test time. Once the optimum bias voltages are determined, they may be permanently programmed into the chip.

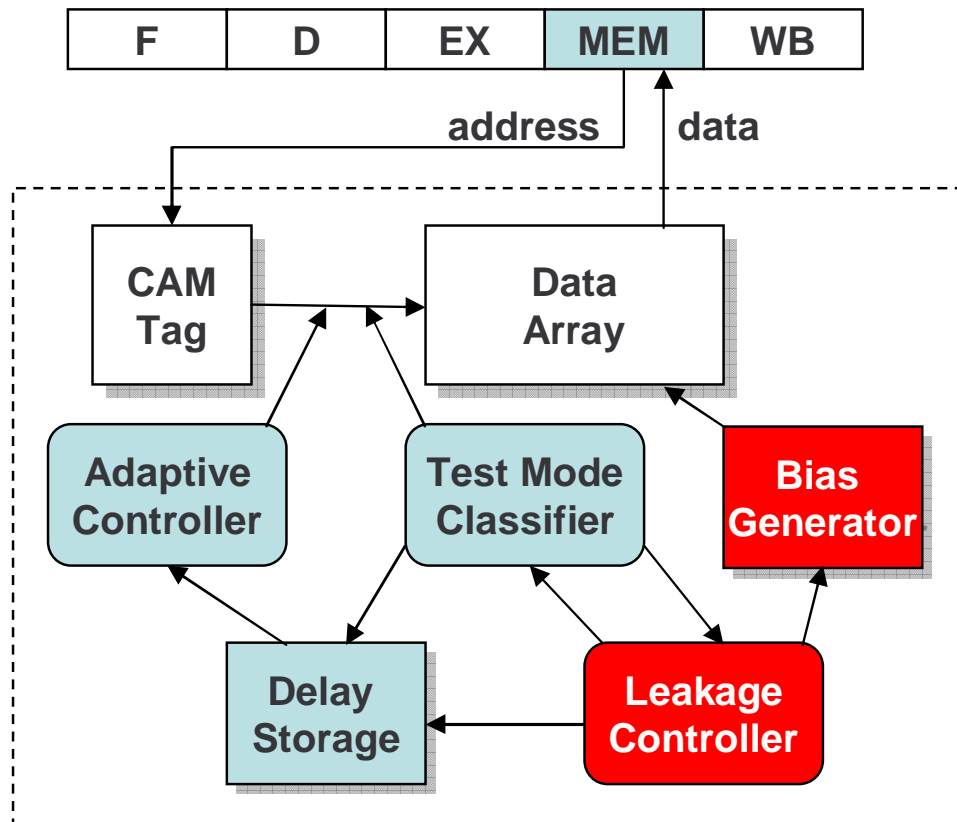


Figure 5.12. The proposed static adaptive cache architecture (shown in a single five-stage pipeline).

One of the important blocks in the static cache architecture is the delay storage unit. This unit stores the speed information and is read along with the data array on every cache access. The operation on the delay storage has two phases: classification and execution.

Figure 5.13 shows the static adaptive cache during the classification phase. The cache is equipped with a leakage controller, which initially tests the entire cache and detects the speed of each cache line by using BIST circuitry for a single NMOS/PMOS body bias

voltage. The leakage controller will increase the threshold voltage of the SRAM cells by using bias generator to minimize the leakage power while meeting the target operating frequency constraints.

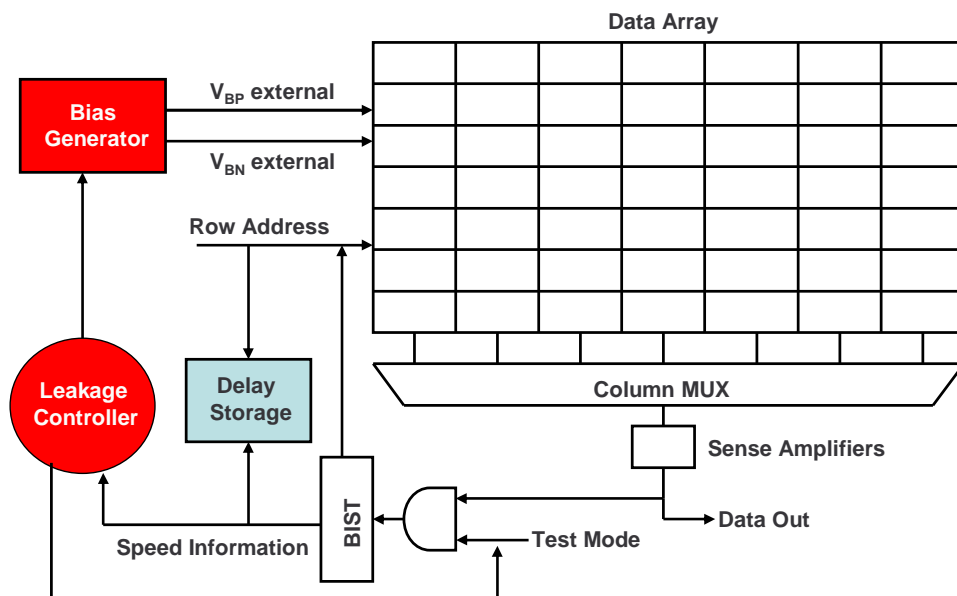


Figure 5.13. The static adaptive cache architecture during the execution phase.

Delay information for each cache line is first achieved during a classification process in which each cache line is probed individually for different body bias voltage and its delay information is written into the delay storage unit. Then, during the execution phase, the delay information is fetched from the delay storage, and each cache line can be accessed based on its estimated speed.

The static adaptive architecture will enable us to minimize power consumption while meeting the operating frequency constraints under the impact of process variations.

Implementing body bias in a chip requires adding power lines for the body bias voltage and including a leakage controller to determine and generate the optimal body bias voltage. The complexity and overhead of the body bias generator and leakage controller circuitry depend on the required resolution in body bias voltage.

The area overhead of body bias was examined by Kuroda and Sakurai [42], who discuss various circuits to apply body bias. The circuitry that controls the body bias is simple, and its area overhead is estimated to be 1% of the chip area. At least two commercial processors use body bias, namely Intel’s Xscale [23] and Transmeta’s Efficeon [29].

5.5 Results and Analysis of Application Performance with an Adaptive Process Resilient Cache Architecture

The static adaptive cache architecture is implemented in SimpleScalar with the simulation parameters summarized in Table 5.7. We have conducted simulations of SPEC2000 benchmarks using the adaptive approach.

Table 5.7. SimpleScalar Parameters for CPU

Parameter	Value
IQ/LSQ/RUU	4/8/16
Fetch, dispatch, commit width	4
Issue width	4, out-of-order
Integer ALU/mult-div	4/1
FP ALU/mult-div	4/1
L1 D-cache Size	16KB, 32-way associative, 32-byte per block (LRU)
L1 I-cache Size	16KB, 32-way associative, 32-byte per block (LRU)
L2 Unified Cache Size	128KB, 64-way associative, 64-byte per block, 8 cycles (LRU)
Memory bus width/latency	8 bytes/100 cycles
Memory Ports	2
I-TLB Size	64-entry, fully associative, 30-cycle miss penalty
D-TLB Size	128-entry, fully associative, 30-cycle miss penalty
Branch Predictor	Combination of bimodal and 2-level gshare; bimodal size 2048; level1 1024 entries, history 10; level2 4096 entries (global)
Branch Target Buffer	512-entry, 4-way associative
Return-address-stack	8-entry

The static adaptive cache delay distributions for different high- V_{th} values in LE SRAM cells are determined by the Monte Carlo simulation, as shown in Table 5.8. In the conservative scheme, we assume all the transistors in the cache to have the same V_{th} , which is equal to 0.23V. Based on our HSPICE simulations, the conservative cache latency equals three cycles for a target frequency of 1GHz. In adaptive cache scheme with no LE SRAM technique, denoted as A1, we assume the transistors have V_{th} equal to 0.2V.

Table 5.8. Distribution of Cache Delay and Leakage Power for Different High- V_{th}

Scheme	V_{th} (V)	Delay (ns)	Mean Leakage (W)	1 cycle	2 cycles	3 cycles
Conservative	0.23	2.34	0.190	0%	0%	100%
A1	0.20	0.952	0.467	75%	24%	1%
A2	0.25	0.972	0.182	68%	30%	2%
A3	0.27	1.091	0.116	56%	40%	4%
A4	0.30	1.122	0.076	45%	50%	5%

In static adaptive cache schemes with LE SRAM technique named A2, A3, and A4, the transistors in the SRAM cell (i.e., not in all the cache circuits) have a high V_{th} of 0.25V, 0.27V, and 0.3V, respectively. In our experiments, we found that the A4 scheme reduces cache mean leakage by more than 6X and decreases the delay by 16% compared to A1. Note that as LE cells are slower than nominal SRAM cells we need to regroup the memory access latencies. This grouping relates to the fraction of accesses that are mapped to one-, two-, three-cycle latencies. These are stored in the delay storage unit and used at runtime to correctly map accesses to various delays. The new grouping is shown in Table 5.8, columns 5, 6, and 7.

We run an exhaustive set of architecture simulations: application performances are shown in Figure 5.14. The comparison is made between a conservative cache that requires three cycles per access and a static adaptive cache that has variable cache access latencies assuming various leakage schemes. Our results show that the static adaptive cache design (with the LE SRAM technique) can achieve a 6% to 16% (on average,

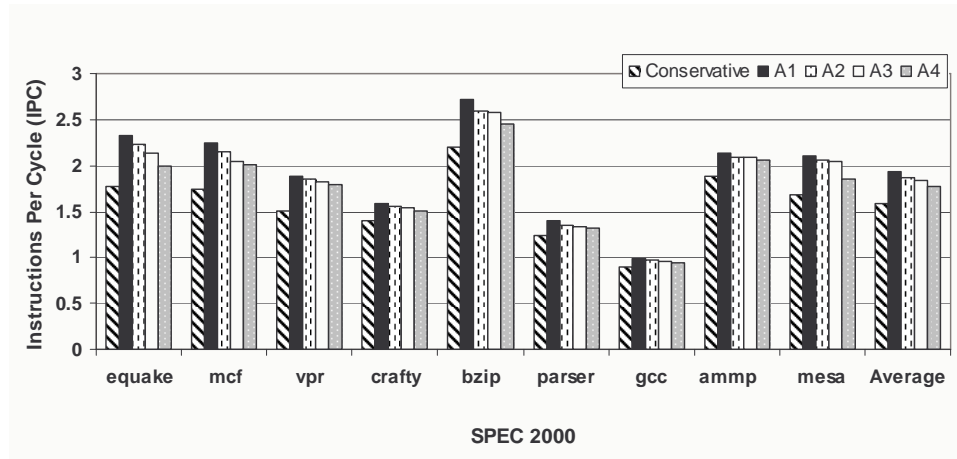


Figure 5.14. Performance improvement between the adaptive caches vs. a conservative cache using three-cycle access time.

10%) performance improvement in the applications studied compared to a conservative design, while providing resilience against failures due to process variations and improving mean leakage power by 6X compared to the adaptive scheme with no leakage optimization technique.

Note that cache leakage optimization can be applied even in a conservative cache but as shown in Chapter 4, the performance would be in that case up to 21% worse compared to the adaptive one. Furthermore, the results shown in figure below for schemes A2, A3, and A4 show that one can control the leakage and performance tradeoffs by choosing the threshold voltages of the SRAM cells. A process variation resilient adaptive cache design can in fact be both performance and power efficient. In our future work, we intend to extend this work with adaptive leakage control through body biasing to maximize performance for a given power budget under process variations.

5.6 Chapter Summary

Process variations will become worse with technology scaling; techniques are necessary at the architecture and circuit levels to mitigate their impact. The mean leakage power could

be impacted by 15X when counting all the possible process parameters. Our static adaptive cache architecture with LE leakage optimizations can improve the application performance in a superscalar design by 6% to 16% (on average, 10%) compared to a conventional design, while providing resilience against process variations and reducing mean leakage power by 6X.

CHAPTER 6

CONCLUSION

In this dissertation, we evaluated the components of a low-power cache suitable for an embedded microprocessor. A comparative study of different cache components using 32-nm PTM technology has been presented with a target clock rate of 1GHz. Energy reduction techniques have been evaluated in the address-decoders, CAM-based tags, SRAM data arrays, and all I/O circuitry. By applying many state-of-the-art low-power techniques, at the circuit level alone, the cache power consumption is reduced by 8X for the same speed and functionality. Moreover, the choice of components and optimizations has a very significant impact on power consumption and is affected by the performance objective and technology node. Our hope is that the methodology used in the dissertation as well as the results and insights presented demonstrate critical issues when designing low-power caches.

Since Process variations will become worse with technology scaling, techniques are necessary at the architecture and circuit levels to reduce the impact of these variations while providing the highest performance for the given power constraints. In this dissertation, we have shown that process variation can have a significant impact on delay (2-3X) under worst-case operating conditions, while under the expected condition a large fraction of accesses would be still close to the nominal value. We have found significant delay variation between worst-case and expected behavioral analysis, motivating us to design an adaptive cache architecture.

The proposed adaptive cache architecture can improve the application performance in a superscalar design by as much as 21% depending on the application and configurations used, compared to a conservative design. This scheme is transparent to other subsystems

and has negligible power and area overhead. The adaptive cache architecture also allows a designer to choose the first-level cache access latency more aggressively and possibly increase the clock rate in a processor design where cache access is the main critical path. Furthermore, The proposed cache architecture could help strike a better balance between power and delay optimizations in a design.

In addition to performance issues, the power consumption of caches can be greatly affected by these variations and therefore a major challenge for processor designers who need to meet power budgets and achieve desired performance levels consistently. The mean leakage power could be impacted by 15X when counting all the possible process parameters. Our static adaptive cache architecture with LE leakage optimizations can, however, improve the application performance in a superscalar design by 6% to 16% (on average, 10%) compared to a conservative design, while providing resilience against process variations and reducing mean leakage power by 6X.

Our hope is that the methodology used in the dissertation as well as the results and insights presented demonstrate critical issues when designing low-power caches in the presence of process variations.

6.1 Future Work

Process variation is a major challenge for processor designers. To address this challenge, we will likely need a combination of solutions at different layers, such as circuits and architecture.

Microprocessors tolerant to process variations in future nanoscale technologies will be at the forefront of innovation for years to come. We believe that the general concept of flexible, variable-latency structures will be applicable to other microarchitectural units in the machine such as the register file and execution units.

In addition to manufacturing variations in gate length and threshold voltage, runtime parameters such as temperature can also have an influence on leakage power. However,

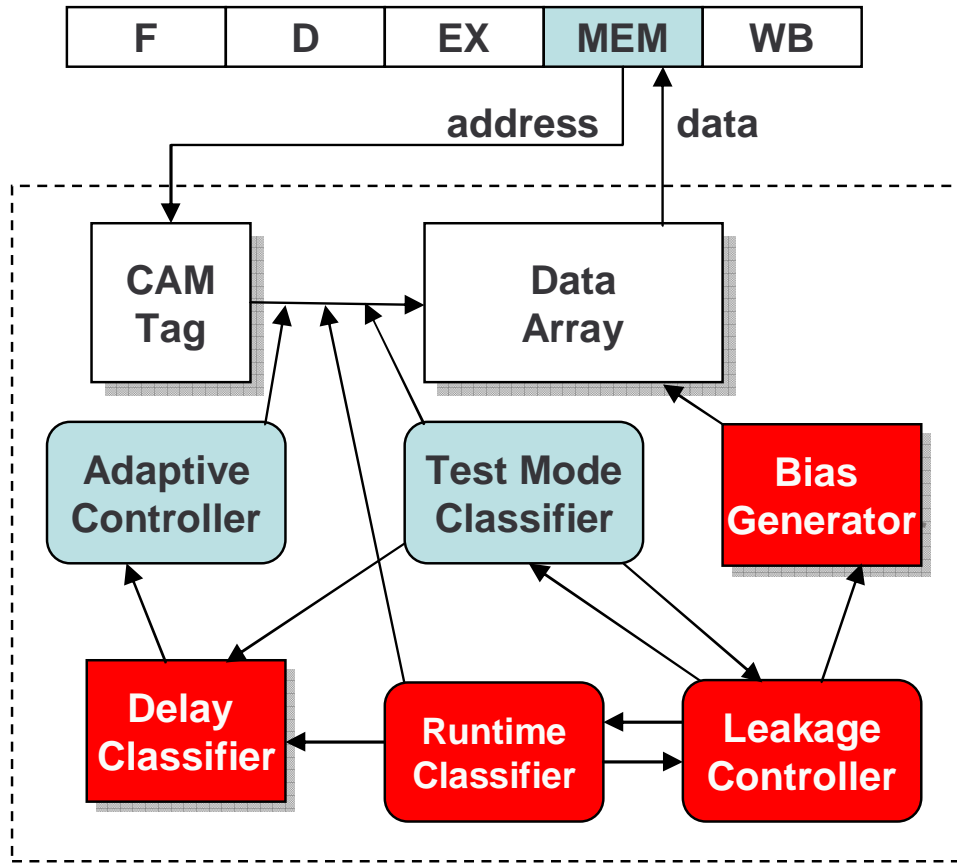


Figure 6.1. The dynamic adaptive cache architecture.

on-chip thermal sensors could be combined with structures such as the delay storage to adjust power estimates for runtime conditions. We identify the following potential future work closely related to the topics of this dissertation.

We are working on dynamic adaptive cache architecture (see Figure 6.1); we call it a dynamic adaptive variable cycle-time cache because the optimum bias voltages are determined dynamically through measurements on our adaptive cache circuit at runtime (adapting to changes in operating conditions such as temperature). To do that, we need to control the threshold voltages for each cache line individually by applying the appropriate amount of body bias voltage.

For example, if the temperature increases, a thermal sensor can be used to detect it during runtime and try to slow down the cache by allowing larger latency. This could happen by using a bias generator to increase the threshold voltages (reverse body bias) of the cache lines that will take one or two cycles, e.g., by changing them to two or three cycles. By doing that, we effectively lower the temperature of the cache and decrease its power consumption.

In addition, if temperature is low and the circuit did not exceed power consumption constraints, the cache performance can be optimized by using bias generator to decrease the threshold voltages (forward body bias) of the cache lines that will take two or three cycles changing them to one or two cycles. The dynamic adaptive cache adjusts the body bias voltages for each cache line dynamically while running an application.

To apply body bias to NMOS, the manufacturing process has to be enhanced with a triple-well process. In a main CMOS process, a p-well in an NMOS can be implanted in a p-type substrate, and thus the biasing of the p-well is common for all the NMOS transistors on the SRAM cell. In order to prevent a global biasing, a triple-well process can be used. The use of a triple-well process provides an opportunity to have a separate biasing on each NMOS transistor and hence adjust a threshold voltage individually on the SRAM cell [57].

BIBLIOGRAPHY

- [1] Predictive technology model. Nanoscale Integration and Modeling Group at ASU. Available Online: <http://www.eas.asu.edu/ptm/>.
- [2] Star-hspice manual, release 2000.4. Synopsys Corporation, December 2000.
- [3] The standard performance evaluation corporation, 2000. Available Online: <http://www.spec.org>.
- [4] Agarwal, Amit, Paul, Bipul C., Mahmoodi, Hamid, Datta, Animesh, and Roy, Kaushik. A process-tolerant cache architecture for improved yield in nanoscale technologies. *IEEE Trans. Very Large Scale Integr. Syst.* 13, 1 (2005), pages 27–38.
- [5] Agarwaland, Aseem, Blaauw, David, and Zolotov, Vladimir. Statistical timing analysis for intra-die process variations with spatial correlations. In *Proceedings of International Conference on Computer Aided Design* (Washington, DC, USA, November 2003), IEEE Computer Society, pp. 900–907.
- [6] Amrutur, B., and Horowitz, M. A replica technique for wordline and sense control in low-power srams. *IEEE Journal of Solid-State Circuits.* Vol. 33, No. 8 (August 1998), pages 1208–1219.
- [7] Ashok, R., Chheda, S., and Moritz, C. Andras. Cool-mem: combining statically speculative memory accessing with selective address translation for energy efficiency. In *ASPLOS* (2002), pp. 133–143.
- [8] Azizi, Navid, Moshovos, Andreas, and Najm, Farid N. Low-leakage asymmetric-cell sram. In *ISLPED '02: Proceedings of the 2002 International Symposium on Low Power Electronics and Design* (New York, NY, USA, 2002), ACM Press, pp. 48–51.
- [9] Bennaser, Mahmoud, Guo, Yao, and Moritz, Csaba Andras. Designing memory subsystems resilient to process variations. In *ISVLSI 07: Proceedings of the IEEE Computer Society Annual Symposium on VLSI* (2007), pp. 357–363.
- [10] Boning, D., and Nassif, S. Models of process variations in device and interconnect, in design of high performance microprocessor circuits, chapter 6, pp. 98-115, IEEE Press, 2000.
- [11] Borkar, Shekhar, Karnik, Tanay, Narendra, Siva, Tschanz, Jim, Keshavarzi, Ali, and De, Vivek. Parameter variations and impact on circuits and microarchitecture. In *DAC '03: Proceedings of the 40th Conference on Design Automation* (New York, NY, USA, June 2003), ACM Press, pp. 338–342.

- [12] Bowman, K., Duvall, S., and Meindl, J. Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration. *IEEE Journal of Solid-State Circuits*. Vol. 37, No. 2 (February 2002), pages 183–190.
- [13] Bowman, K., Tang, Xinghai, Eble, J., and Meindl, James D. Impact of extrinsic and intrinsic parameter fluctuations on cmos circuit performance. *IEEE Journal of Solid-State Circuits*. Vol. 35, No. 8 (August 2000), pages 1186–1193.
- [14] Burger, Douglas C., and Austin, Todd M. The simplescalar tool set, version 2.0. Technical Report CS-TR-1997-1342, University of Wisconsin, Madison, June 1997.
- [15] Burnett, D., Erington, K., Subramanian, C., and Baker, K. Implications of fundamental threshold voltage variations for high-density sram and logic circuits. In *Symposium on VLSI Technology* (June 1994), pp. 14–15.
- [16] Cain, Jason P., and Spanos, Costas J. Electrical linewidth metrology for systematic cd variation characterization and causal analysis. In *Metrology, Inspection, and Process Control for Microlithography XVII*, Daniel J. Herr, Editor, *Proceedings of SPIE* (May 2003), vol. 5038, pp. 350–361.
- [17] Cao, Yun, Gupta, Puneet, Kahng, Andrew, Sylvester, Dennis, and Yang, Jie. Design sensitivities to variability: Extrapolation and assessments in nanometer vlsi. In *IEEE ASIC/SoC Conf.* (September 2002), pp. 411–415.
- [18] Chandra, Saumya, Lahiri, Kanishka, Raghunathan, Anand, and Dey, Sujit. Considering process variations during system-level power analysis. In *ISLPED '06: Proceedings of the 2006 International Symposium on Low Power Electronics and Design* (New York, NY, USA, 2006), ACM, pp. 342–345.
- [19] Chandrakasan, A. Low power circuit and system design, International Electron Device Meeting short course, 2000.
- [20] Chanodia, Itisha, and Velenis, Dimitrios. Effects of parameter variations and crosstalk noise on h-tree clock distribution networks. In *ISVLSI '06: Proceedings of the IEEE Computer Society Annual Symposium on Emerging VLSI Technologies and Architectures* (Washington, DC, USA, 2006), IEEE Computer Society, pp. 456–457.
- [21] Chau, R., Datta, S., Doczy, M., Kavalieros, J., and Metz, M. Gate dielectric scaling for high-performance cmos: from sio₂ to high-k. In *The International Workshop on Gate Insulator* (November 2003), pp. 124–126.
- [22] Chen, Qikai, Mahmoodi, Hamid, Bhunia, Swarup, and Roy, Kaushik. Modeling and testing of sram for new failure mechanisms due to process variations in nanoscale cmos. In *VLSI Test Symposium (VTS)* (May 2005).
- [23] Clark, L., Hoffman, E., Miller, J., Biyani, M., Liao, Y., Strazdus, S., Morrow, M., Velarde, K., and Yarch, M. An embedded 32-b microprocessor core for low-power and high-performance applications. *IEEE Journal of Solid-State Circuits*. Vol. 36 (November 2001), pages 1599–1608.

- [24] Clark, T., Hoffman, J., Miller, J., Biyani, M., Luyun, Liao, Strazdus, Morrow, M., Velarde, E., and Yarch, A. An embedded 32-b microprocessor core for low-power and high-performance applications. In *IEEE Journal of Solid- State Circuits* (November 2001), vol. 36, pp. 1599–1608.
- [25] Cline, Brian, Chopra, Kaviraj, Blaauw, David, and Cao, Yu. Analysis and modeling of cd variation for statistical static timing. In *ICCAD '06: Proceedings of the 2006 IEEE/ACM International Conference on Computer-Aided Design* (New York, NY, USA, 2006), ACM, pp. 60–66.
- [26] Datta, A., Bhunia, S., Mukhopadhyay, S., and Roy, K. Delay modeling and statistical design of pipelined circuit under process variation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 25, 11 (November 2006), 2427–2436.
- [27] Datta, Animesh, Mukhopadhyay, Saibal, Bhunia, Swarup, and Roy, Kaushik. Yield prediction of high performance pipelined circuit with respect to delay failures in sub-100nm technology. In *International On-Line Testing Symposium (IOLTS)* (July 2005).
- [28] Devgan, Anirudh, and Nassif, Sani. Power variability and its impact on design. In *VLSID '05: Proceedings of the 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design (VLSID'05)* (Washington, DC, USA, 2005), IEEE Computer Society, pp. 679–682.
- [29] Ditzel, David R. Power reduction using longrun2 in transmeta's efficeon processor, In Spring Processor Forum, May 2006.
- [30] Efthymiou, Aristides, and Garside, Jim D. An adaptive serial-parallel cam architecture for low-power cache blocks. In *ISLPED '02: Proceedings of the 2002 International Symposium on Low Power Electronics and Design* (New York, NY, USA, 2002), ACM Press, pp. 136–141.
- [31] Eisele, Martin, Berthold, Jorg, Schmitt-Landsiedel, Doris, and Mahnkopf, Reinhard. The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits. *IEEE Trans. Very Large Scale Integr. Syst.* 5, 4 (December 1997), 360–368.
- [32] Furber, S. B., and et al. Arm3 - 32b risc processor with 4kbyte on-chip cache. In *G. Musgrave and U. Lauther, editors, Proc. IFIP TC 10/WG 10.5, Int. Conf. on VLSI* (Elsevier (North Holland), August 1989), pp. 35–44.
- [33] Gowan, Michael K., Biro, Larry L., and Jackson, Daniel B. Power considerations in the design of the alpha 21264 microprocessor. In *Proceedings of the 1998 Conference on Design Automation (DAC-98)* (June 1998), pp. 726–731.
- [34] Hiyatake, H., Tanaka, M., and Mori, Y. A design for high-speed low-power cmos fully parallelcontent-addressable memory macros. *IEEE Journal of Solid-State Circuits*. Vol. 36, No. 6 (June 2001), pages 956–968.

- [35] Humenay, Eric, Tarjan, David, and Skadron, Kevin. Impact of parameter variations on multicore chips. In *Workshop on Architectural Support for Gigascale Integration (ASGI'06)* (June 2006).
- [36] K. Flautner, N. Kim, S. Martin D. Blaauw, and Mudge, T. Drowsy caches: Simple techniques for reducing leakage power. *International Symposium on Computer Architecture* (June 2002).
- [37] Keshavarzi, A., Narendra, S., Bloechel, B., Borkar, S., and De, V. Forward body bias for microprocessors in 130-nm technology generation and beyond. *IEEE Journal of Solid-State Circuits*. Vol. 38, No. 5 (May 2003), pages 696–701.
- [38] Khellah, M., Ye, Y., Kim, N., Somasekhar, D., Pandya, G., Farhang, A., Zhang, K., Webb, C., and De, V. Wordline and bitline pulsing schemes for improving sram cell stability in low-vcc 65nm cmos designs. pp. 9–10.
- [39] Kim, Changkyu, Burger, Doug, and Keckler, Stephen W. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. In *ASPLOS-X: Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems* (New York, NY, USA, 2002), ACM, pp. 211–222.
- [40] Kim, Nam Sung, Kgil, Taeho, Bowman, K., De, V., and Mudge, T. Total power-optimal pipelining and parallel processing under process variations in nanometer technology. In *ICCAD '05: Proceedings of the 2005 IEEE/ACM International Conference on Computer-Aided Design* (Washington, DC, USA, November 2005), IEEE Computer Society, pp. 535–540.
- [41] Kurdahi, Fadi J., Eltawil, Ahmed M., Park, Young-Hwan, Kanj, Rouwaida N., and Nassif, Sani R. System-level sram yield enhancement. In *ISQED '06: Proceedings of the 7th International Symposium on Quality Electronic Design* (Washington, DC, USA, 2006), IEEE Computer Society, pp. 179–184.
- [42] Kuroda, T., and Sakurai, T. Body biasing. in s. narendra and a. chandrakasan, editors, leakage in nanometer cmos technologies, chapter 5, Springer US, 2006.
- [43] Liang, Xiaoyao, and Brooks, David. Latency adaptation for multiported register files to mitigate the impact of process variations. In *ASGI '06: Proceedings of the 2006 Workshop on Architectural Support for Gigascale Integration, held in conjunction with ISCA-33* (June 2006).
- [44] Liang, Xiaoyao, and Brooks, David. Microarchitecture parameter selection to optimize system performance under process variation. In *ICCAD '06: Proceedings of the 2006 IEEE/ACM International Conference on Computer-Aided Design* (New York, NY, USA, 2006), ACM, pp. 429–436.
- [45] Liang, Xiaoyao, and Brooks, David. Mitigating the impact of process variations on processor register files and execution units. In *MICRO 39: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture* (Washington, DC, USA, December 2006), IEEE Computer Society, pp. 504–514.

- [46] Mamidipaka, Mahesh, Khouri, Kamal, Dutt, Nikil, and Abadir, Magdy. Leakage power estimation in srams. Technical Report 03-32, University of California, Irvine, 2003.
- [47] Marculescu, Diana, and Talpes, Emil. Variability and energy awareness: a microarchitecture-level perspective. In *DAC '05: Proceedings of the 42nd annual Conference on Design Automation* (New York, NY, USA, 2005), ACM, pp. 11–16.
- [48] Meng, Ke, and Joseph, Russ. Process variation aware cache leakage management. In *ISLPED '06: Proceedings of the 2006 International Symposium on Low Power Electronics and Design* (New York, NY, USA, 2006), ACM Press, pp. 262–267.
- [49] Mezhiba, A. V., and Friedman, E. G. *Power Distribution Networks in High Speed Integrated Circuits*. Kluwer Academic Publishers, 2004.
- [50] Mohapatra, Debabrata, Karakonstantis, Georgios, and Roy, Kaushik. Low-power process-variation tolerant arithmetic units using input-based elastic clocking. In *ISLPED '07: Proceedings of the 2007 International Symposium on Low Power Electronics and Design* (New York, NY, USA, 2007), ACM, pp. 74–79.
- [51] Montanaro, James, Witek, Richard T., Anne, Krishna, Black, Andrew J., Cooper, Elizabeth M., Dobberpuhl, Daniel W., Donahue, Paul M., Eno, Jim, Hoeppe, Gregory W., Kruckemyer, David, Lee, Thomas H., Lin, Peter C. M., Madden, Liam, Murray, Daniel, Pearce, Mark H., Santhanam, Sribalan, Snyder, Kathryn J., Stephany, Ray, and Thierauf, Stephen C. A 160-MHz, 32-b, 0.5-W CMOS RISC microprocessor. *Digital Technical Journal of Digital Equipment Corporation* 9, 1 (1997).
- [52] Mukhopadhyay, S., Mahmoodi, H., and Roy, K. Modeling of failure probability and statistical design of sram array for yield enhancement in nanoscaled cmos. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 24, 12 (December 2005), 1859–1880.
- [53] Natarajan, A., and Burleson, W. A study of sense amplifiers for advanced microprocessor caches in 70nm technology, Interconnect Circuit Design Group, University of Massachusetts, Amherst.
- [54] Orshansky, Michael, and Keutzer, Kurt. A general probabilistic framework for worst case timing analysis. In *Proceedings of the Design Automation Conference* (New Orleans, LA, USA, June 2002), pp. 556–561.
- [55] Pilo, H., Barwin, J., Braceras, G., Browning, C., Burns, S., Gabric, J., Lamphier, S., Miller, M., Roberts, A., and Towler, F. An sram design in 65nm and 45nm technology nodes featuring read and write-assist circuits to expand operating voltage. pp. 15–16.
- [56] Powell, M., Yang, S.-H., Falsafi, B., Roy, K., and Vijaykumar, T. N. Gated Vdd: A circuit technique to reduce leakage in deep-submicron cache memories. In *ISLPED '00: Proceedings of the 2000 International Symposium on Low Power Electronics and Design* (2000), pp. 90–95.

- [57] Puchner, H., Radaelli, D., and Chatila, A. Alpha-particle seu performance of sram with triple well. *IEEE Transactions on Nuclear Science*. Vol. 51, No. 6 (December 2004), pages 3525–3528.
- [58] Romanescu, Bogdan F., Ozev, Sule, and Sorin, Daniel J. Quantifying the impact of process variability on microprocessor behavior. In *2nd Workshop on Architectural Reliability* (December 2006).
- [59] Schultz, Kenneth J. Content-addressable memory core cells: a survey. *Integr. VLSI J.* 23, 2 (1997), 171–188.
- [60] Tang, Xinghai, De, Vivek K., and Meindl, James D. Intrinsic mosfet parameter fluctuations due to random dopant placement. *IEEE Trans. Very Large Scale Integr. Syst.* Vol. 5, No. 4 (1997), pages 369–376.
- [61] Taur, Yuan, and Ning, Tak H. *Fundamentals of Modern VLSI Devices*. Cambridge University Press, 1998.
- [62] Teodorescu, Radu, Nakano, Jun, Tiwari, Abhishek, and Torrellas, Josep. Mitigating parameter variation with dynamic fine-grain body biasing. In *MICRO 40: Proceedings of the 40th International Symposium on Microarchitecture* (Washington, DC, USA, December 2007), IEEE Computer Society.
- [63] Tiwari, Abhishek, Sarangi, Smruti R., and Torrellas, Josep. Recycle: pipeline adaptation to tolerate process variation. In *ISCA (2007)*, pp. 323–334.
- [64] Tschanz, Jim, Bowman, Keith, and De, Vivek. Variation-tolerant circuits: circuit solutions and techniques. In *DAC '05: Proceedings of the 42nd annual Conference on Design Automation* (New York, NY, USA, 2005), ACM, pp. 762–763.
- [65] Wade, J. P., and Sodini, C. G. Dynamic cross-coupled bit-line content addressable memory cell. *IEEE Journal of Solid-State Circuits*. Vol. 22 (February 1987), pages 119–121.
- [66] Yoshimoto, M., Anami, K., Shinohara, H., Yoshihara, T., Takaji, H., Nakano, S., Kayano, S., and Nakano, T. A 64kb cmos sram with divided wordline structure. *IEEE International Solid State Circuits Conference, Digest of Technical Papers*. Vol. 18 (February 1983), pages 58–59.
- [67] Zhang, M., and Asanovic, K. Highly-associative caches for low-power processors. In *Kool Chips Workshop, 33rd International Symposium on Microarchitecture* (December 2000).