COMPUTER ARCHITECTURE TECHNIQUES FOR POWER-EFFICIENCY

Petitioner Samsung Ex-1032, 0001

Synthesis Lectures on Computer Architecture

Editor

Mark D. Hill, ***University of Wisconsin, Madison***

Synthesis Lectures on Computer Architecture publishes 50 to 150 page publications on topics pertaining to the science and art of designing, analyzing, selecting and interconnecting hardware components to create computers that meet functional, performance and cost goals.

Computer Architecture Techniques for Power-Efficiency
Stefanos Kaxiras and Margaret Martonosi
2008

Chip Mutiprocessor Architecture: Techniques to Improve Throughput and Latency
Kunle Olukotun, Lance Hammond, James Laudon
2007

Transactional Memory
James R. Larus, Ravi Rajwar
2007

Quantum Computing for Computer Architects
Tzvetan S. Metodi, Frederic T. Chong
2006

Petitioner Samsung Ex-1032, 0003

Copyright © 2008 by Morgan & Claypool

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means—electronic, mechanical, photocopy, recording, or any other except for brief quotations in printed reviews, without the prior permission of the publisher.

Computer Architecture Techniques for Power-Efficiency

Stefanos Kaxiras and Margaret Martonosi

www.morganclaypool.com

ISBN: 9781598292084 paper
ISBN: 9781598292091 ebook

DOI: 10.2200/S00119ED1V01Y200805CAC004

A Publication in the Morgan & Claypool Publishers series

SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE #4

Lecture #4

Series Editor: Mark D. Hill, University of Wisconsin, Madison

Library of Congress Cataloging-in-Publication Data

Series ISSN: 1935-3235 print
Series ISSN: 1935-3243 electronic

Petitioner Samsung Ex-1032, 0004

COMPUTER ARCHITECTURE TECHNIQUES FOR POWER-EFFICIENCY

Stefanos Kaxiras
University of Patras, Greece
Kaxiras@ece.upatras.gr

Margaret Martonosi
Princeton University
mrm@princeton.edu

SYNTHESIS LECTURES ON COMPUTER ARCHITECTURE #4

MORGAN & CLAYPOOL PUBLISHERS

Petitioner Samsung Ex-1032, 0005

ABSTRACT

In the last few years, power dissipation has become an important design constraint, on par with performance, in the design of new computer systems. Whereas in the past, the primary job of the computer architect was to translate improvements in operating frequency and transistor count into performance, now power efficiency must be taken into account at every step of the design process.

While for some time, architects have been successful in delivering 40% to 50% annual improvement in processor performance, costs that were previously brushed aside eventually caught up. The most critical of these costs is the inexorable increase in power dissipation and power density in processors. Power dissipation issues have catalyzed new topic areas in computer architecture, resulting in a substantial body of work on more power-efficient architectures. Power dissipation coupled with diminishing performance gains, was also the main cause for the switch from single-core to multi-core architectures and a slowdown in frequency increase.

This book aims to document some of the most important *architectural* techniques that were invented, proposed, and applied to reduce both dynamic power and static power dissipation in processors and memory hierarchies. A significant number of techniques have been proposed for a wide range of situations and this book synthesizes those techniques by focusing on their common characteristics.

KEYWORDS

Computer power consumption, computer energy consumption, low power computer design, computer power efficiency, dynamic power, static power, leakage power, dynamic voltage/frequency scaling, computer architecture, computer hardware.

Petitioner Samsung Ex-1032, 0006

Contents

Acknowledgements xi

1. Introduction 1
   1.1 Brief history of the "power problem" 1
   1.2 CMOS Power Consumption: A Quick Primer 3
      1.2.1 Dynamic Power 3
      1.2.2 Leakage 4
      1.2.3 Other Forms of CMOS Power Dissipation 5
   1.3 Power-Aware Computing Today 5
   1.4 This Book 6

2. Modeling, Simulation, and Measurement 9
   2.1 Metrics 9
   2.2 Modeling basics 11
      2.2.1 Dynamic-power Models 12
      2.2.2 Leakage Models 13
      2.2.3 Thermal models 15
   2.3 Power Simulation 17
   2.4 Measurement 18
      2.4.1 Performance-Counter-based Power and Thermal Estimates 19
      2.4.2 Imaging and Other Techniques 20
   2.5 Summary 21

3. Using Voltage and Frequency Adjustments to Manage Dynamic Power 23
   3.1 Dynamic Voltage and Frequency Scaling: Motivation and Overview 23
      3.1.1 Design Issues and Overview 24
   3.2 System-Level DVFS 26
      3.2.1 Eliminating Idle Time 26
      3.2.2 Discovering and Exploiting Deadlines 28
   3.3 Program-Level DVFS 29
      3.3.1 Offline Compiler Analysis 29
      3.3.2 Online Dynamic Compiler analysis 32
      3.3.3 Coarse-Grained Analysis Based on Power Phases 34

Petitioner Samsung Ex-1032, 0007

3.4 Program-Level DVFS for Multiple-Clock Domains 35
3.4.1 DVFS for MCD Processors 35
3.4.2 Dynamic Work-Steering for MCD Processors 38
3.4.3 DVFS for Multi-Core Processors 40
3.5 Hardware-Level DVFS 41

4. Optimizing Capacitance and Switching Activity to Reduce Dynamic Power 45
4.1 A Road Map for Effective Switched Capacitance 46
4.1.1 Excess Switching Activity 46
4.1.2 Capacitance 49
4.2 Idle-Unit Switching Activity: Clock gating 51
4.2.1 Circuit-Level Basics 51
4.2.2 Precomputation and Guarded Evaluation 53
4.2.3 Deterministic Clock Gating 54
4.2.4 Clock gating examples 56
4.3 Idle-Width Switching Activity: Core 58
4.3.1 Narrow-Width Operands 59
4.3.2 Significance Compression 62
4.3.3 Further Reading on Narrow Width Operands 64
4.4 Idle-Width Switching Activity: Caches 64
4.4.1 Dynamic Zero Compression: Accessing Only Significant Bits 65
4.4.2 Value Compression and the Frequent Value Cache 66
4.4.3 Packing Compressed Cache Lines: Compression Cache and Significance-Compression Cache 68
4.4.4 Instruction Compression 70
4.5 Idle-Capacity Switching Activity 70
4.5.1 The Power-inefficiency of Out-of-order Processors 71
4.5.2 Resource Partitioning 72
4.6 Idle-Capacity Switching Activity: Instruction Queue 75
4.6.1 Physical Resizing 75
4.6.2 Readiness Feedback Control 77
4.6.3 Occupancy Feedback Control 77
4.6.4 Logical Resizing Without Partitioning 78
4.6.5 Other Power Optimizations for the Instruction Queue 80
4.6.6 Related Work on Instruction Windows 81
4.7 Idle-Capacity Switching Activity: Core 82

Petitioner Samsung Ex-1032, 0008

4.8 Idle-Capacity Switching Activity: Caches 84
4.8.1 Trading Memory Between Cache Levels 86
4.8.2 Selective Cache Ways 89
4.8.3 Accounting Cache 91
4.8.4 CAM-Tag Cache Resizing 94
4.8.5 Further Reading on Cache Reconfiguration 97
4.9 Parallel Switching-Activity in Set-Associative Caches 97
4.9.1 Phased Cache 98
4.9.2 Sequentially Accessed Set-Associative Cache 99
4.9.3 Way Prediction 101
4.9.4 Advanced Way-Prediction Mechanisms 104
4.9.5 Way Selection 107
4.9.6 Coherence Protocols 109
4.10 Cacheable Switching Activity 110
4.10.1 Work Reuse 112
4.10.2 Filter Cache 114
4.10.3 Loop Cache 115
4.10.4 Trace Cache 116
4.11 Speculative Activity 117
4.12 Value-dependent Switching Activity: Bus encodings 120
4.12.1 Address Buses 121
4.12.2 Address and Data Buses 122
4.12.3 Further Reading on Data Encoding 124
4.13 Dynamic Work Steering 124

5. Managing Static (Leakage) Power 131
5.1 A Quick Primer on Leakage Power 133
5.1.1 Subthreshold Leakage 134
5.1.2 Gate Leakage 137
5.2 Architectural Techniques Using the Stacking Effect 138
5.2.1 Dynamically Resized (DRI) Cache 139
5.2.2 Cache Decay 141
5.2.3 Adaptive Cache Decay and Adaptive Mode Control 147
5.2.4 Decay in the L2 153
5.2.5 Four-Transistor Memory Cell Decay 155
5.2.6 Gated V_{dd} Approaches for Function Units 156

5.3 Architectural Techniques Using the Drowsy Effect 159
5.3.1 Drowsy Data Caches 159
5.3.2 Drowsy Instruction Caches 161
5.3.3 State Preserving versus No-state Preserving 164
5.3.4 Temperature 167
5.3.5 Reliability 168
5.3.6 Compiler Approaches for Decay and Drowsy Mode 169
5.4 Architectural Techniques Based on V_T 171
5.4.1 Dynamic Approaches 172
5.4.2 Static Approaches 175
5.4.3 Dual-V_T in Function Units 176
5.4.4 Asymmetric Memory Cells 178

6. Conclusions 181
6.1 Dynamic power management via Voltage and Frequency Adjustment: Status and Future Trends 181
6.2 Dynamic Power Reductions based on Effective Capacitance and Activity Factor: Status and Future Trends 182
6.3 Leakage Power Reductions: Status and Future Trends 184
6.4 Final Summary 184

Glossary 187

Bibliography 189

CHAPTER 5

Managing Static (Leakage) Power

Static power consumption has grown to a significant portion of total power consumption in recent years. In CMOS technology, static power consumption is due to the imperfect nature of transistors which "leak" current—thereby constantly consuming power—even when they are not switching. The advent of this form of static power, called *leakage power*, was forecasted early on [32, 136], giving architects the opportunity to propose techniques to address it. Such techniques are the focus of this chapter.

Considerable work to reduce leakage power consumption is taking place at the process level [31]. In fact, process solutions such as the *high-k dielectric materials* in Intel's 45 nm process technology, are already employed. Addressing the problem at the architectural level is, however, indispensable because architectural techniques can be used orthogonally to process technology solutions. The importance of architectural techniques is magnified by the *exponential* dependence of leakage power to various operating parameters such as supply voltage (V_{dd}), temperature (T), and threshold voltage (V_T). Exponential dependence implies that a leakage-reduction solution that works well at some specific operating conditions may not be enough—the problem is bound to reappear with the same intensity as before but at higher temperatures or lower voltages.

Undeniably, the most fruitful ground for developing leakage-reduction techniques at the architectural level has been the cache hierarchy. The large number of transistors in the on-chip memory largely justifies the effort (or obsession) even though these transistors are not the most "leaky"—that distinction goes to the high-speed logic transistors [41]. In addition, the regularity of design and the access properties of the memory system have made it an excellent target for developing high-level policies to fight leakage. Most of the architectural techniques presented in this chapter, therefore, target caches or memory structures.

Chapter structure: The presentation of techniques in this chapter is structured according to the type of low-level leakage-reduction mechanism employed (Table 5.1). Architectural techniques inherit similar characteristics according to the physical quantity that is manipulated by their low-level, leakage-reduction mechanism. Here, we concentrate on three major low-level mechanisms (shown in Table 5.1). The first two, the *stacking effect* and the

Petitioner Samsung Ex-1032, 0011

TABLE 5.1: Structure of the Leakage Reduction Tehniques in this Chapter.

Low Level Mechanism	High-Level Techniques	Characteristics	Section
Stacking effect and gated V_{dd}: sleep transistor cuts off power	Dynamically resized cache (DRI) [239], cache decay [127], adaptive mode control (AMC) [250], functional unit decay [105]	Non-state-preserving (state-destroying) Significant leakage reduction Power-up latency: 10's of cycles	Section 5.2
Drowsy effect: scales supply voltage to reduce leakage	Drowsy caches [77, 137], drowsy instruction caches [138, 139], hybrid approaches (decay + drowsy) [164], temperature-adaptive approaches [129], compiler approaches & hybrids [246]	State-preserving Medium leakage reduction Power-up latency: <10 cycles	Section 5.3
Threshold voltage (V_T) manipulation:	Dynamic Combined V_{dd} (e.g., DVFS) and V_T (e.g., Adaptive Body Biasing—ABB) scaling [163, 231, 70] Static MTCMOS Functional Units [69], Asymmetric Memory Cells [17, 18]	Significant leakage reduction	Section 5.4

Petitioner Samsung Ex-1032, 0012

drowsy mode, manipulate voltage across transistor terminals (source and drain). This affects the magnitude of leakage reduction, the latency in switching leakage modes, and the ability to retain state in the low-leakage mode. The third class of low-level mechanisms manipulates the transistor threshold voltage (V_T) which can dramatically decrease leakage but at the cost of reduced device speed.

It is important to note here that the techniques presented in this chapter address a specific type of leakage, called *subthreshold leakage*. Another type of leakage, called *gate oxide leakage*, is not addressed architecturally but rather at the process level. To gain a better understanding of the structure of this chapter as well as the difference in the two types of leakage, the following section (Section 5.1) delves into the underlying mechanics of leakage.

5.1 A QUICK PRIMER ON LEAKAGE POWER

Static power is so called because it is consumed by every transistor even when no active switching is taking place. In older technologies (e.g., NMOS, TTL, ECL, etc.) it is an inherent problem, because a path from V_{dd} to ground is open even when transistors are not switching. With the advent of CMOS, static power became less of a concern because the *Complementary* gate design prevents open paths from V_{dd} to ground.

Unfortunately, static power resurfaced in CMOS in the form of *leakage power*. In the latest process generations leakage power increases exponentially, principally because of reductions in the threshold voltage. Leakage power increased to levels never seen before in CMOS—levels comparable to the dynamic (switching) power consumption—when technology scaling entered the deep-submicron territory in feature size (<180 nm). Currently, 20–40% of the total power consumption is attributed to leakage power.

CMOS static power arises due to *leakage currents*. The total leakage current (I_{leak}) times the supply voltage gives the static power consumption, P_{leak}:

$$P_{leak} = V \times I_{leak}.$$

Leakage currents are a manifestation of the true *analog* nature of transistors, as opposed to our idealized view of them as perfect digital switches. The state of a transistor (on or off) is controlled by the voltage on its gate terminal. If this voltage is above the *threshold voltage* (V_T) the channel beneath the gate conducts, allowing current in the on state (I_{on}) to flow from the source (V_{dd}) to the drain (GND, ground). In the opposite case (gate voltage below V_T), we like to think that the transistor is off (perfect insulator). But in reality transistors leak: *leakage currents* flow even in their off state. This is evident in the $I-V$ curve where current flows even below the threshold voltage where the device is supposed to be "off."

The current that flows from source to drain when the transistor is off is called *sub-threshold* leakage. But that is not all. There are five more types of leakage: reverse-biased-junction

Petitioner Samsung Ex-1032, 0013

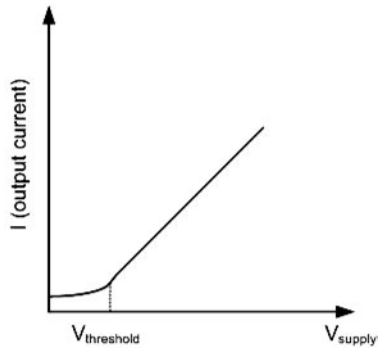

FIGURE 5.1: Example of an "*I–V*" curve for a semiconductor diode (introduced in Chapter 1). Although we informally treat semiconductors as switches, their non-ideal analog behavior leads to leakage currents and other effects.

leakage, gate-induced-drain leakage, gate-oxide leakage, gate-current leakage, and punch-through leakage. The sub-threshold leakage and gate-oxide leakage dominate the total leakage current in devices. Both increase exponentially with each new technology generation with the gate-oxide leakage significantly outpacing the sub-threshold leakage.

In sub-micron technologies, subthreshold and gate leakage is the *cost* we have to pay for the increased speed afforded by scaling. Supply voltage scaling attempts to curb an increase in dynamic power. Unfortunately, this strategy also leads to an enormous increase in the subthreshold and gate leakage problem. This explains why static power has been gaining on dynamic power as a percentage of the total power consumption with every process generation.

5.1.1 Subthreshold Leakage

Subthreshold leakage increases with technology scaling due to V_{dd} scaling. The supply voltage (V_{dd}) is scaled along with other physical quantities to reduce dynamic power consumption. Scaling solely the supply voltage, however, *increases* the *delay* (switching speed) of the transistor. This is because the delay is proportional to the inverse of the current that flows in the *on* state—the I_{on} current (as in the *I–V* curve of Figure 5.1):

$$\text{Delay} \propto \frac{1}{I_{on}} \propto \frac{V_{dd}}{(V_{dd} - V_T)^a}.$$

This current, I_{on}, is a function of the supply voltage and the *difference* between the supply voltage and the threshold voltage (V_T). The factor α is a technology-dependent factor taking values greater than 1 (between 1.2 and 1.6 for recent technologies) [195]. Since V_{dd} is lowered in order to maintain the speed increase from scaling, the only course of action is to also lower the threshold voltage. Herein lies the problem: *subthreshold leakage increases exponentially with lower threshold voltage*.

Petitioner Samsung Ex-1032, 0014

To understand the basic mechanisms for leakage reduction we have to take a closer look at the formulas describing leakage current. We base our discussion on the Berkeley Predictive Model (BSIM3V3.2) formula for subthreshold leakage [143] (which is also the starting point for the simplified Butts and Sohi models [41] discussed in Chapter 2). The formula describing the subthreshold leakage current, I_{Dsub}, is:

$$I_{\mathrm{Dsub}} = I_{s0}\left(1 - e^{\frac{-V_{ds}}{v_t}}\right) e^{\frac{V_{gs}-V_T-V_{off}}{n \cdot v_t}}.$$

Here, V_{ds} is the voltage bias across the drain and the source and V_{gs} is the voltage bias across the gate and source terminal. V_{off} is an empirically determined BSIM model parameter and v_t ($v_t = kT/q$) is a physical parameter called *thermal voltage*[1] which is proportional to the temperature, T. The term n encapsulates various device constants, while the term I_{s0} depends on the transistor geometry (in particular, the aspect ratio of the transistor, W/L).

Immediately, this equation shows the dependence of leakage to W/L, and its *exponential* dependence to V_{ds}, V_{gs}, V_T, and T.

- W/L, *transistor geometry*: Leakage grows with the aspect ratio of a transistor and with its size. Butts and Sohi use simplified models that encapsulate transistor geometry in the k_{design} parameter. They point out that very small transistors such as those found in SRAMs can leak much less than sized-for-performance logic gate transistors. Transistor sizing is primarily a circuit-level concern and it will not preoccupy us at the architecture level.
- V_{ds}, *voltage differential between the drain and the source*: This is probably the most important parameter concerning the architectural techniques developed for leakage. Two important leakage-control techniques that are based on reducing V_{ds} are *the transistor stacking technique*[2] and the *drowsy technique*—a.k.a. dynamic voltage scaling (DVS) for leakage [77]. Both these techniques rely on the $(1 - e^{(-V_{ds}/V_t)})$ factor of the subthreshold leakage equation. This factor is approximately 1 with a large V_{ds} (i.e., $V_{ds} = V_{dd}$ and $V_{dd} \gg v_t$) but falls off rapidly as V_{ds} is reduced. Architectural techniques based on transistor stacking—in particular, a stacking technique called *gated* V_{dd} [184]—and on the drowsy technique form the bulk of the work described in this chapter. The former are presented in Section 5.2 and the latter in Section 5.3.

[1]For the thermal voltage equation, k is Boltzmann's constant and q is the magnitude of the electron's charge. At room temperature ($T = 300$ K), the thermal voltage is about 26 mV.

[2]The stacking effect itself is also partially due to a change in the V_T. This chance is dynamic and is caused by a slight reverse bias induced by the top (off) transistor on the bottom (off) transistor.

Petitioner Samsung Ex-1032, 0015

- V_{gs}, *voltage differential between the gate and source*: Regarding subthreshold leakage for devices in their normal "off" state, this factor can be set to zero, so it is not a concern. Butts and Sohi use this assumption to arrive at their simplified leakage model [41]. However, V_{gs} plays a significant role in the gate-oxide leakage discussed in Section 5.1.2.
- V_T, *threshold voltage*: The threshold voltage—the voltage level that switches on the transistor—significantly affects the magnitude of the leakage current in the *off* state. The exponential dependence of subthreshold leakage on $(V_T)^{-1}$ is evident in the last factor of the BSIM3 formula: the smaller the V_T, the higher is the leakage. Raising the threshold voltage reduces the subthreshold leakage but compromises switching speed.

  Many circuit-level techniques, e.g., MTCMOS, reverse body bias (RBB) and larger-than-V_{dd} forward body bias [13, 174, 14, 222], have been developed to provide a choice of threshold voltages. These techniques provide multiple threshold voltages at the process level (for example, MTCMOS offers high-V_T and low-V_T devices) or vary the threshold voltage dynamically by applying bias voltages on the semiconductor body (e.g., RBB and larger-than-V_{dd} FBB). Architectural techniques based on manipulating the threshold voltage are presented in Section 5.4.
- T, *temperature*: Last but not the least, subthreshold leakage exponentially depends on temperature, T, via the thermal voltage term v_t. This is actually a dangerous dependence since it can set off a phenomenon called *thermal runaway*. If leakage power—or for that matter any other source of power consumption—causes an increase in temperature, the thermal voltage v_t also increases linearly to temperature. This leads, in turn, to an exponential increase in leakage, which further increases temperature. This vicious circle of temperature and leakage increase can be so severe as to seriously damage the semiconductor. The solution is to keep the temperature below some critical threshold so that thermal runaway cannot happen. Cooling techniques, combined with accurate thermal monitoring, are used for this purpose.[3]

  Architecturally, the dependence of leakage to temperature is quite interesting. This is because at low temperatures it might not be so important to engage architectural techniques that could hurt performance with little payoff. As temperature rises and leakage power becomes the dominant component of power consumption (and hence *heat generation*) architectural techniques that can curb leakage become much more appealing. One such example is presented in Section 5.3.4.

[3]Unfortunately, the subject of thermal management, despite its importance, is too extensive to receive other than superficial coverage in the space of this book. Here, it is only mentioned briefly with respect to leakage (Section 5.3.4).

Petitioner Samsung Ex-1032, 0016

5.1.2 Gate Leakage

Gate leakage (also known as gate-*oxide* leakage) is a major concern because of its tremendous rate of increase. It grew 100-fold from the 130 nm technology (2001) to the 90 nm technology (2003) [31]. Major semiconductor companies are switching to *"high-k"* dielectrics in their process technologies to alleviate this problem [31].

Gate leakage occurs due to direct tunneling of electrons through the gate insulator—commonly silicon dioxide, SiO_2—that separates the gate terminal from the transistor channel. The thickness, T_{ox}, of the gate SiO_2 insulator must also be scaled along with other dimensions of the transistor to allow the gate's electric field to effectively control the conductance of the channel. The problem is that when the gate insulator becomes very thin, quantum mechanics allow electrons to *tunnel* across. When the insulating layer is thick, the probability of tunneling across it is virtually non-existent. As the insulating layer becomes thinner, tunneling becomes stronger. Gate-oxide thickness has scaled from 100 nm (1000 Å) to just 1.2 nm (12 Å) in 90 nm and 65 nm technologies. This corresponds to a thickness of just 4–5 atoms [50, 31]! The result is an uncontrollable, exponential increase in gate leakage.

Gate leakage is somewhat dependent on temperature but strongly dependent on the insulator thickness and the gate-to-source (V_{gs}) or gate-to-drain (V_{gd}) biases seen by the device. Without the V_{gs} or V_{gd} biases, the necessary electric field to cause the electrons to tunnel across the gate is absent. Since the supply voltage (V_{dd}) determines the magnitude of V_{gs} and V_{gd}, scaling V_{dd} reduces gate leakage. There is also a weaker dependence of gate leakage on V_{ds}—the voltage across the drain and source—that ties gate leakage to the state of a circuit [190].

The most promising remedy for gate leakage, and the one that is currently in use in the latest generation 45 nm technologies, is to insulate the gate using *high-k* dielectric materials instead of the more common SiO_2 oxide material.[4] A thicker insulating layer of a high-*k* material can be as good as a thin layer of a low-*k* material. The increased thickness significantly reduces the tunneling effect but at the same time does not compromise the ability of the gate to control the channel. In other words, *performance* is not compromised.

Architecturally, gate leakage has not been given the same attention as subthreshold leakage. For the most part, it is considered as an additional leakage component and the hope is that process-level solutions will address the problem. The HotLeakage simulator, mentioned in Chapter 2, takes gate leakage into account, thus giving a more accurate picture for the benefits of various techniques that target subthreshold leakage.

[4]A high dielectric constant, *k*, means that these materials concentrate the electric field better. When used as insulators between the plates of a capacitor, a high dielectric constant yields higher capacitance for the same insulator thickness or, alternatively, the same capacitance but with a thicker insulator layer.

Petitioner Samsung Ex-1032, 0017