EXPLORING EMERGING THREE-DIMENSIONAL INTEGRATION AND MEMORY TECHNOLOGIES IN PROCESSOR MICROARCHITECTURE DESIGN

By

WANGYUAN ZHANG

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2010
To my wife, Wei and my parents
ACKNOWLEDGMENTS

Firstly, I would like to express my deepest gratitude to my advisor, Dr. Tao Li, for his excellent guidance, caring, patience, and providing me with an excellent atmosphere for conducting research. His dedication to work has had a profound influence on both my academic pursuit and my life. I am also grateful for the freedom and the flexibility he gave me throughout my Ph.D. study.

My thanks to the members of my dissertation committee (in alphabetical order), Dr. Randy Chow, Dr. Jose A.B. Fortes, and Dr. Renato J Figueiredo, who have generously given their time and provided invaluable comments to improve my work. I would also like to thank Dr. Lixin Zhang, Dr. Evan Speight, and Dr. Ram Rajamony for providing me the opportunity to work at IBM Austin Research Lab, where I gained valuable experience in microprocessor simulator development and knowledge in microprocessor architecture.

I am grateful to many past and current members of the Intelligent Design of Efficient Architectures Laboratory (IDEAL), including Xin Fu, James M. Poe, Clay Hughes, Changburm Cho. They provided valuable feedback and comments on my research and paper submissions. They shared their research experiences with me to improve my research skills and they helped me to improve my paper writing and presentation skills. I must also acknowledge as well many of my friends whose friendship, hospitality, and knowledge have supported, enlightened, and entertained me over the past few years.

At last, I would like to thank my parents and my wife, Wei Peng. They are always supporting and encouraging me. They have always been cheering me up and standing by my side through the good and bad times.
## TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEDICATION</td>
<td>3</td>
</tr>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>4</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>8</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>9</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>14</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>16</td>
</tr>
<tr>
<td>Using 3D Integration Technology to Enhance Microprocessor Reliability</td>
<td>18</td>
</tr>
<tr>
<td>High performance, Low energy, Reliable, and Durable Memory System Built From Phase-Change Memory Technology</td>
<td>20</td>
</tr>
<tr>
<td>Versatile Cache Architecture Using Non-Volatile SRAM</td>
<td>24</td>
</tr>
<tr>
<td>2 MICROARCHITECTURE VULNERABILITY CHARACTERIZATION AND MİTIGATION UNDER 3D INTEGRATION TECHNOLOGY</td>
<td>27</td>
</tr>
<tr>
<td>3D Microarchitecture Vulnerability Characterization</td>
<td>27</td>
</tr>
<tr>
<td>Radiation Sources</td>
<td>28</td>
</tr>
<tr>
<td>Models for Single Particle Traversing in Matters</td>
<td>28</td>
</tr>
<tr>
<td>Quantitative Effect of 3D Technique on Vulnerability</td>
<td>31</td>
</tr>
<tr>
<td>3D Microarchitecture Reliability Optimizations</td>
<td>35</td>
</tr>
<tr>
<td>Reliability-aware Resource Allocation</td>
<td>35</td>
</tr>
<tr>
<td>Exploring Narrow-Width Operands and ASRAM for Robust 3D Register File</td>
<td>38</td>
</tr>
<tr>
<td>Heterogeneous Integration for Cost-effective Reliability Enhancement</td>
<td>40</td>
</tr>
<tr>
<td>Experimental Methodology</td>
<td>40</td>
</tr>
<tr>
<td>Results</td>
<td>42</td>
</tr>
<tr>
<td>Reliability Improvement due to 3D Shielding Effect</td>
<td>43</td>
</tr>
<tr>
<td>Reliable 3D Register File Design based on Narrow-width Operands</td>
<td>44</td>
</tr>
<tr>
<td>Cost-effective Reliability Optimization by 3D Heterogeneous Integration</td>
<td>45</td>
</tr>
<tr>
<td>Thermal Impact</td>
<td>46</td>
</tr>
<tr>
<td>Related Work</td>
<td>46</td>
</tr>
<tr>
<td>3 EXPLORING PHASE CHANGE MEMORY AND 3D TECHNOLOGY FOR</td>
<td>56</td>
</tr>
<tr>
<td>POWER/THERMAL FRIENDLY, FAST AND DURABLE ON-CHIP DIE-STACKED</td>
<td></td>
</tr>
<tr>
<td>MEMORY ARCHITECTURES</td>
<td></td>
</tr>
<tr>
<td>PCM Power Characterization under 3D Integration Technology</td>
<td>57</td>
</tr>
<tr>
<td>Impact of 3D High-Temperature on PCM power</td>
<td>58</td>
</tr>
<tr>
<td>Impact of 3D TSVs on PCM Power</td>
<td>60</td>
</tr>
<tr>
<td>The Proposed 3D Die-Stacked Hybrid PCM/DRAM System</td>
<td>63</td>
</tr>
<tr>
<td>An Overview of the Hybrid Memory</td>
<td>63</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>PCM-aware OS Paging</td>
<td>65</td>
</tr>
<tr>
<td>Life Span Optimization using Varying ECC Strength and OS-level Wear Leveling</td>
<td>67</td>
</tr>
<tr>
<td>Experimental Methodology</td>
<td>71</td>
</tr>
<tr>
<td>Results</td>
<td>73</td>
</tr>
<tr>
<td>Power Saving of PCM Technology</td>
<td>73</td>
</tr>
<tr>
<td>Endurance Enhancement</td>
<td>76</td>
</tr>
<tr>
<td>Thermal Relief and Performance Benefit</td>
<td>77</td>
</tr>
<tr>
<td>Memory-Intensive Workload Results</td>
<td>78</td>
</tr>
<tr>
<td>Related Work</td>
<td>79</td>
</tr>
<tr>
<td>4 CHARACTERIZING AND MITIGATING THE IMPACT OF PROCESS VARIATIONS ON PHASE CHANGE BASED MEMORY SYSTEMS</td>
<td>89</td>
</tr>
<tr>
<td>Process Variations in PCM: The Source and Impact</td>
<td>90</td>
</tr>
<tr>
<td>PCM Design Parameters subject to Process Variation</td>
<td>90</td>
</tr>
<tr>
<td>Characterizing the Effect of Process Variations on PCM Programming Current</td>
<td>92</td>
</tr>
<tr>
<td>PV Induced PCM Programming Power Overhead and Reliability</td>
<td>95</td>
</tr>
<tr>
<td>Mitigating Process Variation Impact on PCM Power and Reliability</td>
<td>96</td>
</tr>
<tr>
<td>Variation-aware Programming Current Provision</td>
<td>96</td>
</tr>
<tr>
<td>Adaptive Data Comparison Write Using Page Classification</td>
<td>98</td>
</tr>
<tr>
<td>Adaptive Memory Compression with PCM Cell Refreshing</td>
<td>101</td>
</tr>
<tr>
<td>Experimental Methodology</td>
<td>105</td>
</tr>
<tr>
<td>Results</td>
<td>108</td>
</tr>
<tr>
<td>Power Savings</td>
<td>108</td>
</tr>
<tr>
<td>Endurance Enhancement</td>
<td>110</td>
</tr>
<tr>
<td>Sensitivity Analysis of Tuning Resolution</td>
<td>111</td>
</tr>
<tr>
<td>Related Work</td>
<td>111</td>
</tr>
<tr>
<td>5 A RESISTANCE DRIFT RESILIENT ARCHITECTURE FOR MULTI-LEVEL CELL PHASE CHANGE MEMORY SYSTEM</td>
<td>120</td>
</tr>
<tr>
<td>Background: Multi-level Cell Phase Change Memory (MLC-PCM) and Resistance Drift</td>
<td>121</td>
</tr>
<tr>
<td>Multi-level Cell Phase Change Memory</td>
<td>121</td>
</tr>
<tr>
<td>Resistance Drift and its Impact on MLC</td>
<td>122</td>
</tr>
<tr>
<td>Resistance Drift on MLC-PCM System: Characterization and Implication</td>
<td>125</td>
</tr>
<tr>
<td>A Characterization of Resistance Drift Dynamics on MLC-PCM System</td>
<td>125</td>
</tr>
<tr>
<td>Implication of Conservative Resistance Drift Tolerance on PCM Power and Endurance</td>
<td>128</td>
</tr>
<tr>
<td>Helmet: Hardening Resistance Drift with Minimized Power and Endurance Impact</td>
<td>130</td>
</tr>
<tr>
<td>Data Inversion and Rotation</td>
<td>130</td>
</tr>
<tr>
<td>Hybrid SLC/MLC</td>
<td>134</td>
</tr>
<tr>
<td>Temperature Aware Page Allocation</td>
<td>137</td>
</tr>
<tr>
<td>Experimental Methodology</td>
<td>140</td>
</tr>
<tr>
<td>Results</td>
<td>142</td>
</tr>
<tr>
<td>Efficiency of Drifting Tolerance</td>
<td>142</td>
</tr>
<tr>
<td>Sensitivity Analysis</td>
<td>146</td>
</tr>
<tr>
<td>Related Work</td>
<td>147</td>
</tr>
</tbody>
</table>
ENHANCING THE VERSATILITY, FUNCTIONALITY AND EFFICIENCY OF MULTI-CORE CACHE ARCHITECTURE USING NON-VOLATILE MEMORY BASED CELL DESIGN

Magnetic Tunnel Junction (MTJ) based Non-Volatile Memory

Non-Volatile SRAM (NV-SRAM)

NV-SRAM Cell: Organization and Operation Modes

Performance, Power and Area Characterization of NV-SRAM based Cache

Exploring the Benefits of NV-SRAM based Caches

Reducing Context Switch Overhead due to Cache Interference

Reducing Cache Leakage Power

Utilizing NV-SRAM to Reduce Pre-fetching-induced Cache Pollution

Achieving Versatile Functionalities Simultaneously through Page Coloring-based Cache Management

Experimental Methodology

Results

Performance Improvement due to the Reduced Context-switch Induced Interferences

Power Saving and Reliability Enhancement

Performance Improvement on Pre-fetching

Simultaneous Cache Optimizations across Performance, Power, and Reliability Domains

Related Work

CONCLUSIONS

REFERENCE LIST

BIOGRAPHICAL SKETCH
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>A Summary of parameters in modeling a striking of alpha particle on silicon</td>
<td>49</td>
</tr>
<tr>
<td>2-2</td>
<td>Comparison of Qcrit and SERs for ASRAM and SRAM</td>
<td>49</td>
</tr>
<tr>
<td>2-3</td>
<td>Base configuration</td>
<td>49</td>
</tr>
<tr>
<td>2-4</td>
<td>Thermal parameters used in thermal analysis</td>
<td>50</td>
</tr>
<tr>
<td>3-1</td>
<td>Baseline machine configuration</td>
<td>81</td>
</tr>
<tr>
<td>3-2</td>
<td>Workloads</td>
<td>81</td>
</tr>
<tr>
<td>3-3</td>
<td>DRAM/PCM timing and power parameters</td>
<td>82</td>
</tr>
<tr>
<td>3-4</td>
<td>The impact of PCM technology trend on power and performance (Normalized to baseline)</td>
<td>82</td>
</tr>
<tr>
<td>4-1</td>
<td>Baseline machine configuration</td>
<td>113</td>
</tr>
<tr>
<td>4-2</td>
<td>PCM timing and power parameters</td>
<td>113</td>
</tr>
<tr>
<td>4-3</td>
<td>Benchmarks used to form workloads</td>
<td>113</td>
</tr>
<tr>
<td>4-4</td>
<td>Workloads</td>
<td>114</td>
</tr>
<tr>
<td>5-1</td>
<td>Baseline machine configuration</td>
<td>149</td>
</tr>
<tr>
<td>5-2</td>
<td>MLC PCM parameters</td>
<td>149</td>
</tr>
<tr>
<td>5-3</td>
<td>Benchmarks used to form workloads</td>
<td>150</td>
</tr>
<tr>
<td>5-4</td>
<td>Workloads</td>
<td>150</td>
</tr>
<tr>
<td>5-5</td>
<td>A comparison of normalized power and endurance between two schemes that achieves the same error rate</td>
<td>151</td>
</tr>
<tr>
<td>6-1</td>
<td>Area, performance and power comparison between SRAM and NV-SRAM based caches</td>
<td>187</td>
</tr>
<tr>
<td>6-2</td>
<td>Baseline machine configuration</td>
<td>187</td>
</tr>
<tr>
<td>6-3</td>
<td>Benchmarks</td>
<td>188</td>
</tr>
<tr>
<td>6-4</td>
<td>Workloads</td>
<td>188</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>Energy loss of alpha particle in silicon.</td>
<td>51</td>
</tr>
<tr>
<td>2-2</td>
<td>Possession energy of alpha particle during striking process.</td>
<td>51</td>
</tr>
<tr>
<td>2-3</td>
<td>A cross-section view of the simulated 3D processor.</td>
<td>51</td>
</tr>
<tr>
<td>2-4</td>
<td>Energy spectrum of alpha particle and the % of particles reaching active layers on each die.</td>
<td>52</td>
</tr>
<tr>
<td>2-5</td>
<td>Reliability-aware instruction scheduling and dynamic promotion in 3D issue queue.</td>
<td>52</td>
</tr>
<tr>
<td>2-6</td>
<td>Segmented 3D ROB design for reliability-aware scheduling and promotion.</td>
<td>52</td>
</tr>
<tr>
<td>2-7</td>
<td>Narrow width value and ASRAM based 3D register file design.</td>
<td>53</td>
</tr>
<tr>
<td>2-8</td>
<td>An overview of 3D die stacked processor SER estimation.</td>
<td>53</td>
</tr>
<tr>
<td>2-9</td>
<td>Circuit simulation of particle strike on SRAM.</td>
<td>53</td>
</tr>
<tr>
<td>2-10</td>
<td>The floor-plans of simulated 2D and 3D processors.</td>
<td>53</td>
</tr>
<tr>
<td>2-11</td>
<td>Normalized SER of 3D implementation.</td>
<td>54</td>
</tr>
<tr>
<td>2-12</td>
<td>A) The IQ reliability improvement achieved by scheduling and promotion technique. B) The ROB reliability improvement achieved by scheduling and promotion technique. C) The LSQ reliability improvement achieved by scheduling and promotion technique.</td>
<td>54</td>
</tr>
<tr>
<td>2-13</td>
<td>The impact of reliable register file design on SER.</td>
<td>55</td>
</tr>
<tr>
<td>2-14</td>
<td>The impact of reliable register file design on IPC.</td>
<td>55</td>
</tr>
<tr>
<td>2-15</td>
<td>SER comparison of different SOI integration schemes under 3D technologies.</td>
<td>55</td>
</tr>
<tr>
<td>2-16</td>
<td>Chip temperature of (1) the baseline 3D microarchitecture design and (2) 3D microarchitecture with reliability optimizations.</td>
<td>55</td>
</tr>
<tr>
<td>3-1</td>
<td>The basic structure of a PCM cell*</td>
<td>83</td>
</tr>
<tr>
<td>3-2</td>
<td>The programming pulses of PCM[21]</td>
<td>83</td>
</tr>
<tr>
<td>3-3</td>
<td>Sub-array architecture of PCM memory.</td>
<td>83</td>
</tr>
<tr>
<td>3-4</td>
<td>The overall architecture of a PCM device and the illustration of a successful RESET and the one dimensional temperature profile [74]</td>
<td>84</td>
</tr>
<tr>
<td>Page</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>3-5</td>
<td>The temperature dependence of a PCM device’s programming power.</td>
<td></td>
</tr>
<tr>
<td>3-6</td>
<td>2D planar PCM prototype.</td>
<td></td>
</tr>
<tr>
<td>3-7</td>
<td>PCM constructed in true-3D organization.</td>
<td></td>
</tr>
<tr>
<td>3-8</td>
<td>A) An overview of the proposed hybrid PCM/DRAM memory system architecture and B) The 3D integration of processor and memory system.</td>
<td></td>
</tr>
<tr>
<td>3-9</td>
<td>Latency overhead with the increased number of corrected errors.</td>
<td></td>
</tr>
<tr>
<td>3-10</td>
<td>Wear-out aware OS page allocation.</td>
<td></td>
</tr>
<tr>
<td>3-11</td>
<td>A) The Cross-section view of the simulated 3D (Left) and B) the floor-plan of processor layer (unit mm) (Right).</td>
<td></td>
</tr>
<tr>
<td>3-12</td>
<td>A breakdown of memory power (workload: H4).</td>
<td></td>
</tr>
<tr>
<td>3-13</td>
<td>The percentage of pages migrated to DRAM.</td>
<td></td>
</tr>
<tr>
<td>3-14</td>
<td>A comparison of overall memory power.</td>
<td></td>
</tr>
<tr>
<td>3-15</td>
<td>The performance impact of PCM memory design.</td>
<td></td>
</tr>
<tr>
<td>3-16</td>
<td>A comparison of endurance improvement.</td>
<td></td>
</tr>
<tr>
<td>3-17</td>
<td>The impact of wear-out aware page allocation on BCH performance.</td>
<td></td>
</tr>
<tr>
<td>3-18</td>
<td>An illustration of on-chip temperature (workload: H3).</td>
<td></td>
</tr>
<tr>
<td>3-19</td>
<td>An example of thermal constraint on maximal allowed frequency (workload: H3).</td>
<td></td>
</tr>
<tr>
<td>3-20</td>
<td>The peak temperature reduction due to the PCM’s power reduction and the execution speedup enabled by the reduced peak temperature.</td>
<td></td>
</tr>
<tr>
<td>3-21</td>
<td>PCM power impact for memory-intensive workloads.</td>
<td></td>
</tr>
<tr>
<td>3-22</td>
<td>PCM performance impact for memory-intensive workloads.</td>
<td></td>
</tr>
<tr>
<td>4-5</td>
<td>The electrical paths of PCM write operations.</td>
<td></td>
</tr>
<tr>
<td>4-6</td>
<td>Variation in the width of $I_{RESET_{min}}$ due to transistors’ length variation.</td>
<td></td>
</tr>
<tr>
<td>4-7</td>
<td>Variation in the amplitude of $I_{RESET}$ delivered to PCM cell due to transistors’ length variation.</td>
<td></td>
</tr>
<tr>
<td>4-8</td>
<td>Variation in the amplitude of $I_{RESET_{min}}$ due to BECD variation.</td>
<td></td>
</tr>
<tr>
<td>4-9</td>
<td>Variation in the amplitude of $I_{RESET_{min}}$ due to ThickHEATER variation.</td>
<td></td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>4-10</td>
<td>Variation in the amplitude of $I_{\text{RESET, min}}$ due to ThickGST variation.</td>
<td>116</td>
</tr>
<tr>
<td>4-11</td>
<td>The color map and distribution of minimum required programming current for a PCM chip</td>
<td>116</td>
</tr>
<tr>
<td>4-12</td>
<td>The cycle lifetime as a function of programming pulse energy [24].</td>
<td>117</td>
</tr>
<tr>
<td>4-13</td>
<td>Fine-grained voltage tuning for write driver circuits.</td>
<td>117</td>
</tr>
<tr>
<td>4-14</td>
<td>(A) The implementation of adaptive data comparison write (ADCW) and (B) OS-level page classification.</td>
<td>117</td>
</tr>
<tr>
<td>4-15</td>
<td>A profiling of the number of dirty words in the evicted cache lines.</td>
<td>118</td>
</tr>
<tr>
<td>4-16</td>
<td>Adaptive memory compression.</td>
<td>118</td>
</tr>
<tr>
<td>4-17</td>
<td>Line shifting mechanism.</td>
<td>118</td>
</tr>
<tr>
<td>4-18</td>
<td>A comparison of write power saving.</td>
<td>118</td>
</tr>
<tr>
<td>4-19</td>
<td>The performance impact of variation-aware PCM design.</td>
<td>119</td>
</tr>
<tr>
<td>4-20</td>
<td>Power consumption distribution.</td>
<td>119</td>
</tr>
<tr>
<td>4-21</td>
<td>A comparison of endurance improvement.</td>
<td>119</td>
</tr>
<tr>
<td>4-22</td>
<td>Sensitive analysis of tuning resolution.</td>
<td>119</td>
</tr>
<tr>
<td>5-1</td>
<td>An illustration of the MLC programming mechanism[8].</td>
<td>152</td>
</tr>
<tr>
<td>5-2</td>
<td>(A) Resistance drift of a PCM device over time (B) Readout reliability issue due to resistance drift (C) Addressing read reliability via increasing margin.</td>
<td>152</td>
</tr>
<tr>
<td>5-3</td>
<td>The dependence of read reliability on resistance margin between two adjacent states for the low-temperature off-chip design and high-temperature 3D on-chip design.</td>
<td>152</td>
</tr>
<tr>
<td>5-4</td>
<td>The dependence of (A)power consumption and (B)endurance on resistance margin.</td>
<td>153</td>
</tr>
<tr>
<td>5-5</td>
<td>Using bit- inversion, rotation or the combination of both to convert the original data to drift-tolerant one.</td>
<td>153</td>
</tr>
<tr>
<td>5-6</td>
<td>An overview of microarchitectural support for data inversion and rotation.</td>
<td>153</td>
</tr>
<tr>
<td>5-7</td>
<td>An overview of the thermal sensor network and temperature-aware page allocation.</td>
<td>154</td>
</tr>
<tr>
<td>5-8</td>
<td>A comparison of absolute readout error rate.</td>
<td>154</td>
</tr>
<tr>
<td>5-9</td>
<td>The breakdown percentage of state “01” and “10” before and after applying inversion and rotation.</td>
<td>154</td>
</tr>
</tbody>
</table>
5-10 The percentage of pages that operate in the SLC mode when the 0uA margin is used...154
5-11 The impact of reduced error rate on performance ........................................................155
5-12 A comparison of power consumption............................................................................155
5-13 A comparison of endurance..........................................................................................155
5-14 Sensitive analysis of inversion/rotation granularity......................................................155
6-1 (A) Parallel state (state “0”) (B) Anti-parallel state (state “1”). ..............................189
6-2 The structure of a NV-SRAM cell..................................................................................189
6-3 NV-SRAM operation modes (V_{high} : full supply voltage level , V_{low} : zero supply voltage level ).................................................................189
6-4 HSPICE simulation waveforms of NV-SRAM RECALL operation.............................190
7-6 (A) 3D-structre of a NV-SRAM cell (B) Layout of a conventional SRAM cell (C) Layout of a NV-SRAM cell..................................................190
6-6 NV-SRAM based cache architecture..............................................................................191
6-7 The use of NV-SRAM to reduce cache interference misses........................................191
6-8 The illustration of cases in which NV-SRAM cache can improvement pre-fetching performance. ........................................................................................................191
6-9 The tag array of decoupled NV-SRAM Cache..............................................................192
6-10 An illustration of the page coloring technique............................................................192
6-11 Intra-partition-based simultaneous versatility..........................................................192
6-12 Inter-partition-based simultaneous versatility.........................................................193
6-13 Performance improvement due to the reduced interference......................................193
6-14 (A)Performance improvement comparison due to the reduced context switch overhead on four-threaded multiprogramming workloads (B) Percentage of time that each application is allowed to use the non-volatile storage by NV-SRAM caches with QoS. .........................................193
6-15 A comparison of overall L2 cache energy.................................................................194
6-16 A comparison of the soft error rate of L2 cache.......................................................194
6-17  (A) The efficiency of pre-fetching on NV-SRAM based caches (B) the entering impact

6-18  The efficiency of simultaneous versatile functionalities.
Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

EXPLORING EMERGING THREE-DIMENSIONAL INTEGRATION AND MEMORY
TECHNOLOGIES IN PROCESSOR MICROARCHITECTURE DESIGN

By

Wangyuan Zhang

August 2010

Chair: Tao Li
Major: Electrical and Computer Engineering

Aggressive scaling of process technologies has allowed the semiconductor industry to keep pace with Moore’s Law for the past several decades. However, CMOS process technology is approaching its limits and interconnects are becoming a major performance bottleneck. Moreover, microprocessor designers are facing an increasing number of related challenges, including high power consumption, low reliability, enlarged performance gap between high-speed processor and off-chip memory, and increased demand for high-density memory. In response to these issues, new devices and manufacturing process technologies have been proposed. Among them, three-dimensional (3D) integration is a promising technology for extending Moore’s Law by stacking multiple layers of processed silicon with very high-density, low-latency, and vertical interconnects. Phase Change Memory (PCM) is another emerging technology, which is regarded as a promising candidate for the next generation of computer memory and may help solve the power and reliability challenges faced by designers. However, these emerging technologies pose unanswered questions to the field of computer architecture: What are the impacts of these emerging technologies on the microarchitecture design? How can these resources be leveraged effectively to design future processor innovatively? What new challenges are introduced and how can they be addressed?
To attack these questions, this research adopts a multi-layer holistic approach, including investigating the characteristics of emerging technologies from a microarchitecture perspective at the device and circuit level; proposing innovative architectures that explore the benefits and mitigate design issues at the microarchitecture level; and designing OS schemes at the system level for better interaction between OS and microarchitecture. The first task in this work is a study of the reliability benefits of 3D integration. The study reveals opportunities for leveraging the structure of vertical stacking and the heterogeneous process technologies to achieve enhanced reliability. Next, a memory architecture built from PCM is proposed to achieve scalable, reliable, high performance, low power, and thermal friendly features for both conventional 2D microprocessor system and die-stacked microarchitecture in the upcoming 3D-stacking era. To effectively integrate PCM into a conventional memory hierarchy, microarchitecture- and system-level techniques are proposed to combat several issues of PCM-based memory, including long write latency, large programming power consumption, high susceptibility to process variation, and low reliability. Further, the application of new memory technologies on cache architecture is investigated. Versatile cache architecture is proposed that employs non-volatile SRAM cell design, which integrates new memory devices into conventional SRAM cells. This new design opens new opportunities to improve the efficiency of cache management as well as enabling versatile and enriched functionalities for cache architecture.
CHAPTER 1
INTRODUCTION

Three-dimensional (3D) integration is an emerging fabrication technology that increases transistor density by vertically stacking multiple dies and connecting the stacked dies with a high-density, low latency die-to-die interconnect interface. The result of 3D die stacking is a significant reduction of interconnect length both within a die and across dies in a system, which translates into power savings or increased performance. Additional benefits of 3D integration include a higher packing density, smaller footprint due to a third dimension, greater flexibility in routing signals, the ability to integrate disparate fabrication technologies, and the potential for new microarchitecture organizations. Due to these benefits, 3D integration has become of critical interest to the semiconductor community. From a microarchitecture design standpoint, 3D integration provides many new opportunities. For example, as more and more transistors are packed on a chip, design of fault tolerant microprocessor systems continues to face further challenges. Transient faults, also known as soft errors, are increasingly becoming a reliability concern to high-performance microprocessor fabricated using state-of-the-art CMOS technologies. Although the performance and power benefits of 3D have been extensively investigated recently [1][2][3][4][5], the reliability implications of using 3D technology is largely unknown. In this work, the reliability benefit of using 3D technology is investigated for the first time. The analysis reveals opportunities for leveraging 3D vertical stacking and heterogeneous integration capability to cost-effectively enhance the microarchitecture resilience to soft error strikes.

Another benefit of 3D technology is its application on memory system design to attack the Memory Wall problem, which refers to the growing disparity of speed between CPU and off-chip memory. An important reason for this disparity is the limited communication bandwidth and
high latency beyond chip boundaries. 3D integration provides an excellent solution to address the issue by enabling die-stacked on-chip memory, which is bounded to processor chip with low-latency and high-bandwidth vertical interconnects. Despite the promising advantage of 3D stacking, there is a significant concern for its thermal impact, since die stacking can dramatically increase power density substantially. As processor designers are moving toward the direction of die-stacked on-chip DRAM to alleviate memory latency and bandwidth issues, elevated on-chip temperatures [1][2] present significant challenges for on-chip DRAM power management [6]. This is because the charge leakage of a DRAM cell grows exponentially as the temperature increases and it causes increased DRAM power overhead in refreshing. Subsequently, the further increased power can potentially exacerbate existing 3D thermal problems. Furthermore, main memory consisting entirely of DRAM is already hitting power and cost limits. Exploiting emerging memory technologies, such as Phase Change Memory (PCM), has become crucial to build larger capacity, power efficiency, and thermal friendly memory systems in the future. In this work, memory architecture build from PCM, as a replacement for DRAM, is proposed. Microarchitecture- and system- level techniques are proposed to overcome the drawbacks of PCM, including high latency, limited lifespan, and low reliability.

This work takes one step further to explore the application of emerging memory technologies on cache architecture design and proposes a novel, versatile cache architecture that employs Non-Volatile SRAM (NV-SRAM) cell design. The proposed cell design integrates new memory devices into standard SRAM cells, which opens new opportunities to improve the performance, power, and efficiency of cache architectures by enabling versatile and enriched functionality, such as reducing context-switching induced cache interference misses, reducing cache leakage power, improving the efficiency of prefetching schemes.
These three aspects of the study are described in details in the rest of this chapter.

**Using 3D Integration Technology to Enhance Microprocessor Reliability**

Semiconductor transient faults are an emerging reliability threat to high-performance microprocessors [7]. Transient faults, also known as soft errors or Single Event Upsets (SEUs), are caused by cosmic rays or alpha particles that can potentially corrupt program data, leading to erroneous computation results. Although these faults do not permanently damage the circuit, the incorrect execution of programs cannot be tolerated. As CMOS processing technologies continue to scale down, the soft error rate on future microprocessors is expected to grow rapidly.

In a microprocessor with 2D planar implementation, energetic particles have a shallow, unobstructed path to reach the active surface of the silicon chip. Under 3D integration technologies, stacking multiple dies on top of each other suggests that the incident particles need to penetrate through multiple layers of material before they can strike transistors on the inner layers. Whether the stacked dies have the capability of intercepting particles before they reach deep into the 3D chip and how this could change the Soft Error Rate (SER) across different layers of the 3D chip are largely unknown. In this research, comprehensive analytical models that capture the heterogeneous transient fault resilient characteristics due to the effect of die stacking are presented. Combing circuit and microarchitecture soft error susceptibility analysis, the characterization of 3D microarchitecture soft error vulnerability of microprocessors with die stacking implementations is performed. This study reveals opportunities for enhancing the 3D microarchitecture reliability to soft error:

“The structure of vertical stacking”: Vertically stacking dies is a unique feature of 3D integration. This study shows that going vertically makes the outer-dies shield particle strikes for the inner-dies. For instance, it is observed that more than 90% of incident alpha-particles can be stopped by the outer-dies, thereby leading to a heterogeneous error rate across layers in 3D
microarchitecture. As the SER is proportional to the flux of incident particles, stopping most particles by the outermost layer reduces the SERs of inner layers by up to 90%.

“The capability of heterogeneous integration”: In the past, a great number of radiation-hardened circuits and devices have been proposed to improve soft error reliability. Many are prevented from wide deployment due to their implementation overhead. With the heterogeneous integration capability offered by 3D technologies, the cost-effective radiation-hardening 3D microarchitecture design that exploits hybrid building circuits is investigated. For example, a conventional 2D chip can be implemented on a Silicon-On-Insulator (SOI) wafer, which is more resilient to particle strikes due to its limited sensitive volume of charge collection. However, the SOI wafer adds to the manufacturing cost of 10-15% [8]. By applying the SOI technology only at the most vulnerable layer in a 4-layer 3D microarchitecture design, the desired reliability target can be achieved while eliminating a significant portion (75%) of SOI cost.

Microarchitecture design optimizations are proposed in this work that can take advantage of the aforementioned 3D integration benefit on soft error reliability. The techniques are built on top of previously proposed 3D microarchitecture design [1][9], i.e. partitioning each microarchitecture component and distributing partitions across different layers. At the microarchitecture level, program execution states can be classified into the Architecturally Correct Execution (ACE) bits [10] on which soft error strikes will result in visible errors in the final program outcomes and into un-ACE bits which do not contribute to correct program execution. By mapping ACE bits to the robust layers while leaving un-ACE bits on the vulnerable layers, microarchitecture soft error susceptibility can be significantly reduced, since the vulnerable program states are protected by the 3D’s shielding effect. Moreover, the heterogeneous integration capability of 3D technologies allows reliability-hardening circuit
design to be applied selectively on the vulnerable layers while leveraging the vertical die stacking effect to protect other layers. It is shown that the reliability of 3D register files can be cost-effectively enhanced by deploying SEU resilient Asymmetric SRAM (ASRAM) circuits to the vulnerable layer where the most significant bits of register values are mapped while leveraging the 3D shielding effect to protect other bits that reside in the inner layers. In addition, thermal analysis is performed to investigate the impact of the proposed techniques on chip temperature which is a major concern of 3D microarchitecture design.

**High performance, Low energy, Reliable, and Durable Memory System Built from Phase-Change Memory Technology**

Dynamic Random Access Memory (DRAM) has been used as the main memory in computer systems for decades due to its high-density, high-performance and low-cost. However, DRAM technologies are facing both scalability and power issues. DRAM is difficult to scale down beyond 50nm [11] due to various limitations associated with device leakages and retention time. DRAM-based main memory is also consuming an increasing proportion of the power budget and has been reported to account for as much as 40% of the total system power [12].

3D integration is a promising technology to help combat the “Memory Wall” in the future multi-core processors. Previous studies [1][2][13] have observed impressive performance improvement by stacking memory on top of the microprocessor with low-delay, high-bandwidth connections between them. Despite the performance advantage, the elevated on-chip temperature [1][2] due to high power density presents significant challenges for DRAM power management [6]. Since the charge leakage of a DRAM cell grows exponentially as the temperature increases, the elevated on-chip temperature will accelerate the degradation of DRAM data retention, which needs to be addressed by increasing DRAM refresh frequency. Consequently, the 3D-stacked DRAM is expected to operate at double (or higher) the current refresh rate [6], leading to a
higher refresh power overhead. Worse, this increased power dissipation will further aggravate on-chip temperature and exacerbate thermal constraints. To achieve low power, traditional DRAM power management techniques attempt to eliminate the unnecessary refreshes [6][14] or put idle banks into power saving mode [15]. However, the temperature dependent DRAM leakage cannot be overcome.

In contrast to conventional memory technology, a cutting-edge memory technology, PCM is attracting increasing attention as a promising candidate for next generation memories [16][17][18][19]. PCM is a type of non-volatile memory that uses the unique behavior of chalcogenide glass, which can be switched between two states (i.e. crystalline and amorphous) with the application of heat. The desirable characteristics of PCM include random access, fast read access, low standby power, superior scalability (no physical limits down to 20nm technology node [20]), compatible with CMOS process[21] etc. In this research, a 3D-stacked memory system built from PCM, as a replacement for DRAM, with lower power consumption and alleviated temperature constraints is proposed. The approach leverages two attractive features provided by PCM: low standby power and high-temperature friendly operation. The former is a common feature of all non-volatile storages as data can be retained in them even when not powered. The latter is a unique characteristic of PCM: to store data in PCM, the temperature needs to be elevated to switch the state of cells. Using analytical and circuit-level modeling that characterizes PCM in detail, it is observed that the programming power of PCM cells can be reduced as the chip temperature is elevated, in contrast to the exponential increase in refresh power for DRAM when chip temperature increases. This high-temperature friendly feature makes PCM superior to DRAM for die-stacked memory systems. In addition, it is investigated that the power benefit of using 3D TSVs to deliver PCM programming current.
Compared with conventional metal wires, the low resistance of TSVs minimizes the dissipated power along the bit lines. It is shown that PCM benefits more from TSVs than DRAM in terms of power savings.

Despite its superior features, there are several challenges to overcome before PCM can be integrated into memory architecture. First, PCM is much slower than DRAM, especially for write operations, thereby adversely impacting system performance. Second, PCM devices can only sustain a limited number of write, which is significantly smaller than DRAM, therefore the write traffic to these devices must be reduced. Otherwise, the short lifetime may significantly limit the usefulness of PCM. Third, the superior scalability of PCM is one of its most attractive features and allows PCM to scale down for the next several generations of processing technology. Nevertheless, difficulties in manufacturing control have resulted in significant variability in the fabricated devices. The system-level implication of process variation on PCM memory has largely unexplored. Finally, the continued quest for high integration density has recently motivated the PCM R&D community to design and fabricate PCM devices with multi-level cells (MLC) [102][103][104]. To store more bits in a single PCM cell, the MLC-PCM relies on fine-grained resistance partitioning, leading to a reduced margin that separates the states. As PCM relies on the phase change material’s resistance to represent stored information, its meta-stable nature represents a concern for PCM to ensure data correctness during the read operations.

To address the issues of write latency and limited endurance, a hybrid memory design that is composed of a large portion of PCM used as a primary memory space and a small portion of DRAM that serves as a write buffer to reduce the number of writes to the PCM partition. To maximize the runtime efficiency of the DRAM-based write buffer, an OS-level paging scheme that takes into account the memory reference characteristics of applications and migrates the hot-
modified pages from PCM to DRAM so that the life time degradation of PCM is alleviated. As writes to the PCM cannot be entirely avoided and the wear-out on a given PCM cell is exacerbated as the number of overwrites increases, it is crucial to provide an endurance optimization scheme that can maximize and balance the life span of all PCM cells. Toward this end, a synergetic reliability enhancement approach that combines architecture- and OS-level wear-out optimizations. At the architecture level, the proposed approach uses Error Correction Code (ECC) with varied strength to detect and correct a number of errors in each memory block fetched. At the OS-level, an OS page allocation scheme is proposed to perform endurance-aware allocation. For the process variation issue, major sources of process variation in PCM design are identified and the impacts of PCM parameters variability on power and endurance are quantified in this work. It is shown that the worst-case-driven design methodology used in conventional design compromises the power benefit and exacerbates the wear-out of PCM. Cross-layer approaches are explored in this work to mitigate these overheads, including circuit-level adaptation technique; microarchitecture-level data-comparison-write and memory compressions; and OS-level page classification and adaptation schemes. As for the reliability concern of MLC-PCM, unwanted resistance drift of PCM devices is characterized for both off-chip and 3D die-stacked memory design. The analysis shows that such resistance drifts leads to a substantial increase in readout error rate. To address this issue, a conventional design adopts a large margin between two adjacent states to guard against drift, but at the cost of high power consumption and low endurance. Several techniques are proposed in this work to relax the requirement on margin size, while preserving the readout reliability, such as bit-flip and rotation operation to convert drift-sensitive data patterns to drift-tolerant ones, hybrid SLC/MLC-PCM design that allows MLC to operate in the SLC mode, and temperature-aware OS page allocation.
Versatile Cache Architecture Using Non-Volatile SRAM

Typically, caches are implemented using SRAM due to its fast access speed. To represent the binary values, SRAM stores charges. When information stored in the SRAM cells is destroyed, data has to be retrieved from the low-level in memory hierarchy, resulting in significant performance penalty and power overhead. Moreover, as CMOS processing technologies scale down, the leakage-induced stand-by power of SRAM becomes an increasing concern. In the past, numerous techniques have been proposed to optimize the efficiency of SRAM-based caches. These techniques, however, often target on optimizing a specific aspect of the cache design. Since performance, power and reliability are all converging as important design challenges; it becomes more crucial to explore cache architectures that can flexibly support design optimizations in more than one domain. Moreover, in light of multi-/many- core computing and virtualization, the processor concurrently executes a large number of threads/applications with drastically different characteristics. This trend demands a versatile and flexible cache substrate that allows multiple caching functionalities and optimizations to be simultaneously triggered at run-time. The conventional cache architectures are incapable of achieving the above goals. In this work, a versatile cache architectures is proposed that can simultaneously enhance the performance, power and reliability of caches while allowing the co-existing of multiple and versatile caching functions and optimizations.

Emerging memory technologies open new opportunities for cache optimizations. For example,[77][117] propose hybrid cache architectures consisting of SRAM and non-volatile memories to improve the performance and power efficiency of caches. In their hybrid cache design, the non-volatile memory cells and the SRAM cells are used separately and in isolation. In other words, levels of cache hierarchies are built from disparate memory technologies, but cells in a single hierarchical level or bank are designed using the same memory technology. This
work moves one step further by advocating a novel, Non-Volatile SRAM (NV-SRAM) design that synergically integrates Magnetic Tunnel Junction (MTJ) based non-volatile memory devices into the standard SRAM cells. The proposed NV-SRAM cell design consists of a conventional six-transistor SRAM cell, two MTJs stacked on top of the SRAM and two extra transistors. The MTJs are connected to the SRAM partition through the two extra access transistors and they serve as backup storage for the SRAM cell or additional storage besides the SRAM cell, depending on the operation mode. When MTJs are used as backup storage, the proposed NV-SRAM design allows data stored in the SRAM to be copied into the coupled non-volatile devices by a store operation and the stored value to be restored back to the SRAM by a recall operation. Alternatively, when the MTJs are used as additional storage, the NV-SRAM design allows two different values to be stored into SRAM and MTJs separately. Compared to the conventional SRAM and non-volatile memory cells, the proposed NV-SRAM provides several benefits: (1) Unlike non-volatile memory, the read and write to the NV-SRAM remain high-speed (e.g. 10-50X faster than non-volatile memories), which is inherited from SRAM; (2) Unlike SRAM, the NV-SRAM offers non-volatility, which stems from the characteristics of non-volatile memory devices and allows data to be preserved without consuming leakage power; (3) The NV-SRAM increases effective storage capacity due to the capability of recording different data in non-volatile memory storage and SRAM storage simultaneously; (4) The tight coupling of the SRAM and the non-volatile devices in each cell facilitates the data transfer between them and allows instance backup and recovery of cache data, thereby avoiding long latency of data refilling from the lower level of memory hierarchies. Consequently, cache architectures built using NV-SRAM not only share the high-speed characteristics of conventional SRAM but also provide versatile
and enriched features, including power-saving, non-volatility and instant data recovering, which make it suitable for a wide range of optimizations.

For the single domain optimizations, NV-SRAM can effectively reduce cache interference and performance overhead due to context switching. The basic idea is to utilize the non-volatile elements in NV-SRAM cells as additional storage capacity to perform instant backing up and restoring cache lines when context switching occurs. NV-SRAM can also be used to achieve leakage power reduction by backing up data into the non-volatile elements and completely powering off cache lines. The additional storage capacity provided by non-volatile elements can be utilized to reduce cache pollution due to aggressive or useless pre-fetching. For simultaneous multiple domain optimization, architecture and operating system support are proposed to achieve simultaneous versatility. The proposed design consists of two components: an OS-level page coloring mechanism for cache partition, and a hardware mechanism that allows the operating mode of each NV-SRAM cache partition to be configured independently. With this cache partition and independent configuration capability, different features can be enabled at different regions of the NV-SRAM cache, depending on which aspect of the application accessing this partition needs to be optimized.
CHAPTER 2
MICROARCHITECTURE VULNERABILITY CHARACTERIZATION AND MITIGATION UNDER 3D INTEGRATION TECHNOLOGY

Both the performance benefits and the thermal impact of 3D die stacked microarchitecture have been studied recently [1][2][3][4][5], but the reliability implication of using 3D technology in processor microarchitecture design has received little or no attention. In a microprocessor with 2D planar implementation, energetic particles have a shallow, unobstructed path to reach the active surface of the silicon chip. Under 3D integration technologies, stacking multiple dies on top of each other suggests that the incident particles need to penetrate through multiple layers of material before they can strike transistors on the inner layers. Whether the stacked dies have the capability of intercepting particles before they reach deep into the 3D chip and how this could change the soft error rate (SER) across different layers of the 3D chip are largely unknown. This research makes the first attempt to study the microarchitecture vulnerability of 3D chips. Comprehensive analytical models that capture the heterogeneous transient fault resilient characteristics due to the effect of die stacking are presented in this chapter. Combing circuit and microarchitecture soft error susceptibility analysis, 3D microarchitecture soft error vulnerability of microprocessors with die stacking implementations is characterized. The study reveals opportunities for enhancing the 3D microarchitecture reliability to soft error:

3D Microarchitecture Vulnerability Characterization

In this section, a physical model that captures one particle traversing materials is first described. Then this model is used to analyze a flux of particles striking a 3D chip to illustrate the shielding effect of vertically stacked structures. Furthermore, reliability-hardening techniques (e.g. asymmetric SRAM and SOI) enabled by 3D heterogeneous integration is discussed and their benefits are evaluated on reliable 3D microarchitecture design.
**Radiation Sources**

There are two primary mechanisms by which the aforementioned ionization radiation can release charge in a semiconductor device and cause soft errors: (1) direct ionization by the incident particle itself and (2) ionization by secondary particles created by nuclear reactions between the incident particle and struck device. Research and study [24] in past decades have found that at sea level, SEU in semiconductor devices is induced by two primary radiation sources: alpha particles and high-energy neutrons from cosmic radiation.

There are two sources of alpha-particle emissions: interconnect metallization and package material. Alpha particles emitted from radioactive impurities (such as 232Th and 238U) present in the IC plastic packaging material is the primary particle sources of an emission rate at about [25], while the metallization layer is generally a small contributor to soft error with an estimated emission rate of [26]. An emitted alpha particle is capable of ionizing silicon by generating electron-hold pairs. In contrast to alpha particle, neutrons do not directly produce charges in silicon, but it can interact with silicon and the reaction products are able to generate charges by ionizing silicon. At the sea level, alpha-induced soft error is a major source of the total failures [27]. Therefore, this study focuses on alpha-induced soft error in this paper. As impurities in package are a strong source of alpha particles, great effort has been spent on material improvement. A factor of ten-time reduction in flux at the chip surface may be eventually realized [28]. Although this level is low in magnitude, it is still too high from the point of reliable operation, especially considering the aggressive scaling trend of transistor processing techniques.

**Models for Single Particle Traversing in Matters**

When an energetic particle hits a device and starts passing through semiconductor materials, ionization radiation takes place as the incident particle collides with silicon lattices. In this ionization process, the incident particle gradually loses its energy due to collisions with
atoms in the lattice and will finally come to rest inside the semiconductor materials if it doesn’t possess enough energy to penetrate the chip. The energy loss due to interacting with materials determines the amount of charge generated as well as the maximal distance this particle can travel. Understanding the aforementioned ionization process will help us analyze how many dies of a 3D chip are struck by an incident particle and the amount of charge deposited on each layer.

As a particle passes through layers of 3D chip, its energy loss per unit path length can be quantified by the linear energy transfer (LET) metric [29]. Therefore, the total amount of energy used for charge generation can be expressed by Equation 2-1, where $E_{\text{deposite}}$ is the energy deposited by a striking particle traveling a distance of $s$, and $\rho$ is the density of the material. The energy required to generate an electron-hole pair is defined by $w_{\text{ehp}}$ ($w_{\text{ehp}}$ for various materials can be found in [30]). Equation 2-2 derives the charge deposited during this ionization process, where $q = 1.6022 \times 10^{-19}$ Coulombs/e.

$$E_{\text{deposite}} = \text{LET} \cdot \rho \cdot s$$  \hspace{1cm} (2-1)

$$Q_{\text{deposite}} = E_{\text{dep}} \cdot q / w_{\text{ehp}}$$  \hspace{1cm} (2-2)

Combining Eq 2-1 and Eq 2-2, we obtain:

$$Q_{\text{deposite}} = \text{LET} \cdot \rho \cdot s \cdot q / w_{\text{ehp}}$$  \hspace{1cm} (2-3)

In Equation 2-3, LET can be either obtained from radiation experiments or expressed mathematically by the Bethe-Bloch formula $-dE/dx$ [31] shown as Equation 2-4.

$$-\frac{dE}{dx} = K \rho \frac{Z z^2}{A} \left[ \frac{1}{2} \ln \left( \frac{2m c^2 \beta^2 \gamma^4 T_{\text{max}}}{\gamma^2} \right) - \beta^2 \right]$$  \hspace{1cm} (2-4)

As LET is represented by $-dE/dx$, substituting LET in Equation 2-3 with its mathematical formula shown in Equation 2-4, we obtain

$$Q_{\text{deposite}} = K \rho \frac{Z z^2}{A} \left[ \frac{1}{2} \ln \left( \frac{2m c^2 \beta^2 \gamma^4 T_{\text{max}}}{\gamma^2} \right) - \beta^2 \right] \cdot \rho \cdot s \cdot q / w_{\text{ehp}}$$  \hspace{1cm} (2-5)
Equation 2-4 and Equation 2-5 can be used for analyzing the interaction between incident particles and each stacked die in a 3D IC. In particular, Equation 2-4 allows us to measure the maximal distance that a particle can penetrate with given initial energy. Therefore, the number of dies along the vertical direction that will be affected by this particle can be determined. Equation 2-5 describes the charge deposited by a particle with respect to the distance that it has traveled. Consequently, it can be used for estimating the charge deposited at a specific layer of the multiple stacked dies. To quantify the amount of charge deposited and the distance an alpha particle can travel in a 3D chip, analytical methods are built based on Equation 2-4 and Equation 2-5 that model the striking process of an alpha particle hitting silicon. These models take the initial energy value of alpha particles as an input and compute the energy loss for each $\Delta x$ distance traveled as well as the energy remaining in the incident particle after traveling a distance of $x$. Table 2-1 summarizes the model parameters and their values.

At the ground level, alpha particles in packaging material typically possess a kinetic energy between 4MeV and 10MeV [32]. Therefore, the penetration process is simulated by sweeping initial energy across this range with a step size of 1MeV. Figures 2-1 and 2-2 show the energy loss and the trend of $Q_{\text{deposit}}$ (since $Q_{\text{deposit}} \propto -dE/dx$) with respect to the distance an alpha particle travels into silicon and the energy left of the incident particle. As can be seen, the energy loss rate, shown as Figure 2-1, increases as the particle’s velocity decreases. This is because decreased traveling speed allows more time for the interaction with the material. This corresponds to an increase of the generated charge as the incident particle slows down. Figure 2-2 shows that alpha particles with a typical energy range of 4MeV to 10MeV have the capability of passing through a distance between 20μm and 70μm into a chip. Given the typical thickness of each die is around 30μm, this implies that most alpha particles are stopped in the two outer-
layers. In the following section, how the SER varies due to the flux reduction across layers will be quantified.

**Quantitative Effect of 3D Technique on Vulnerability**

This research focuses on the SER of SRAM-based sequential and memory circuits because the SER of combinational logic is currently much lower [33]. The reason for that is an error in a combinational logic circuit can be efficiently masked before being latched. At the circuit level, the SER can be expressed by the following model [34]

\[
SER_{\text{circuit}} \propto F \cdot (A_{d,p} + A_{d,a}) \cdot K \cdot e^{-\frac{Q_{\text{crit}}}{Q_s}}
\]

where \( F \) represents particle flux, \( K \) denotes an overall scaling factor. \( Q_{\text{crit}} \) is the critical charge which represents the minimum amount of charge required to flip a storage node and generate a soft-error. \( (A_{d,p} + A_{d,a}) \) describes the diffusion area sensitive to particle strikes. It implies that the probability of a soft error is linearly related to the probability that a particle hits the sensitive drain area. The charge collection efficiency \( (Q_s) \) is a measure of the amount of deposited charge that a circuit node collects after a particle impact. Equation 2-6 shows several factors that affect SER substantially. The main attributes of interest are 3D’s impact on \( F \) and \( Q_{\text{crit}} \). To quantify the impact of 3D technologies or 3D-enabled schemes on these factors, the parameters are normalized to those of a baseline 2D processor. There are several methods for die stacking and the wafer-to-wafer bonding [1][2] is assumed in this research, which is the mainstream processing technique used in industry [2]. Figure 2-3 shows a cross section view of the studied 3D processor with four stacked dies. Each layer is a die manufactured using a typical planar technology with thinned bulk silicon, active devices and metal layers. Through Silicon Vias are etched through substrate to bond layers together.
The 3D Shielding Effect on Particle Flux

As discussed above, the striking particle losses energy while traveling through the absorber material. The distance that an incident particle can travel is limited by its initial kinetic energy. The typical energy of an alpha particle emitted by radioactive impurities falls between 4MeV and 10MeV and this implies a travel distance between 20μm to 70μm into the chip (see Figure 2-2). In a 3D die stacking structure, particles with lower energy are likely to come to rest before reaching inner-layers. As a result, the stacked organization helps reduce the particle flux (the major factor in determining the amount of deposited charges) to the inner layers, leading to heterogeneous SER across dies.

To quantify $F$ and the consequent SER for each layer, a 3D die stacked microprocessor is considered and it is shown in Figure 2-3. Figure 2-3 also provides a summary of the physical dimension of each stacked layer. The thickness of each die is around 27μm with 2μm spacing between two dies [1]. As discussed previously, both the package material and the metal layer contain impurities that produce particles. The particle flux emitted by the package material is approximately 9 times higher than by the metal layer. $F$ is used to denote the total alpha particle flux in a planar chip and the portions contributed by the package and metal layer are $0.9F$ and $0.1F$ respectively. An alpha radioactive source with 50/50 U/Th mix is simulated by closely resembling the energy spectrum based on experimental results presented in [29][35]. The energy spectrum is shown in Figure 2-4. For the packaging emission source, only particles emitted from the large surfaces of the package are considered, which are parallel to the chip. This is because the probability of emitted alpha particles from surrounding packing material from directions which are vertical to the die is significantly smaller than from directions which are parallel to the dies. It is assumed that all emitted particles strike the active device surface from a vertical
direction with a zero angle in order to estimate the maximal distance they can penetrate into the chip. Using the analytical models developed previously, the percentage of emitted particles that affect each layer is derived. Figure 2-4 shows that particles are attenuated as they move through the chip layers: (1) 37.4% of particles emitted from bottom package layer come rest before even striking the transistors on die 1, while less than 0.4% of these particles are able to reach to die 2 and beyond. Alpha particles emitted from the top package need to pass through the bulk silicon several hundred micro meters in thickness before striking transistors on die 4. Due to the limited energy possessed by alpha particles, the thick bulk silicon is able to stop all particles produced by the top package layer. (2) Impurity sources in the metal layer affect each layer equally because a metal layer exits in each die. As shown in Figure 2-4, among all particles emitted from the package, 37.4% of them stop in the metal layer of die 1 and the remaining 62.6% of the particle flux reaches the active layer of die 1. However, only 0.4% of the particle flux is able to reach to the active layer of die 2. Using the flux distribution between the two radiation sources, the aggregated flux for each die in a 4-layer 3D chip is derived as follows:

\[
F_{2D} = F_{3D\_Die\_1} = 0.1F \cdot 62.6\% + 0.9F \cdot 62.6\% = 62.6\%F \quad \Rightarrow \quad SER_{2D} = SER_{3D\_Die\_1} = SER_{raw}
\]

\[
F_{3D\_Die\_2} = 0.1F \cdot 62.6\% + 0.9F \cdot 0.4\% = 6.27\%F \quad \Rightarrow \quad SER_{3D\_Die\_2} = 0.1SER_{raw}
\]

\[
F_{3D\_Die\_3,4} = 0.1F \cdot 62.6\% = 6.26\%F \quad \Rightarrow \quad SER_{3D\_Die\_3,4} = 0.1SER_{raw}
\]

As the SER is proportional to the alpha particle flux and we assume a \(SER_{raw}\) for the baseline 2D processor, the soft error rate of die 1 (the bottom layer) in the 3D chip is \(SER_{raw}\) while other layers all have an error rate of 10% (i.e. 6.27%/62.6%) of \(SER_{raw}\).

The shielding effect of 3D opens a wide range of opportunities for cost-effective soft error reliability optimizations. For instance, since it is orthogonal to most existing reliability optimization techniques, it can be combined with conventional fault tolerance mechanisms that target 2D planar chips. Moreover, the shielding effect offers a wide range of protection for all
types of circuits including storage cell, sequential circuits and combinational logic. In addition, since the shielding effect provides protection for multiple dies simultaneously; reliability-hardening techniques only need to be selectively deployed at the vulnerable layers. In the following section, microarchitecture design that exploits the benefit of 3D shielding effect on reliability is proposed.

**3D-enabled Heterogeneous Processing Techniques for Reducing Vulnerability**

3D integration’s capability of stacking dies vertically allows the realization of heterogeneous integration. Optimized circuit design with different performance, power and reliability characteristics can be built on different layers. This research shows a reliable register file design by using reliability-optimized ASRAM circuitry on selective layers (detailed later in this chapter). The ASRAM [36] is composed of asymmetric transistors. It improves stability by selecting a preferred state and weakening those transistors necessary to dramatically change the state of the cell. This weakness is achieved by appropriately increasing transistor sizes. Thus, this cell has very low SER as compared to the standard SRAM cell, but at the cost of layout area required. ASRAM circuits can be adopted on layers where the vulnerability is high, while traditional SRAM can be placed on other layers where compact layout is required. Moreover, the circuit can be built with different processes or even on different types of wafers. This ability to provide mixed-process integration can also have significant impact on reliability-aware microarchitecture design. Bulk silicon technology is a conventional method used for fabricating microprocessors. With bulk silicon technology, charge generated along the path of the incident particle can be collected up to several microns deep within the silicon substrate. Recently, silicon on insulator (SOI) has emerged as an alternative to bulk-CMOS due to the many advantages offered by this new technique. Among them, the SER benefit of SOI devices is very attractive for 3D IC design. With SOI technologies, charge collection only involves the thin silicon layers
above the buried oxide, while bulk silicon can collect charge from much greater depth. As a result, the SOI devices collect less charge from radiation events. This feature suggests the SER advantage of SOI technologies. Similarly, SOI can be applied on selective dies depending on the reliability requirements in those layers.

3D Microarchitecture Reliability Optimizations

This section presents techniques that can cost-effectively optimize 3D microarchitecture soft error robustness by exploiting the die stacking reliability benefit that is observed in the previous section.

Reliability-aware Resource Allocation

Microarchitecture structures such as the issue queue (IQ), reorder buffer (ROB) and load store queue (LSQ) hold information from in-flight instructions that is crucial for correct program execution. Compared to 2D planar microarchitecture, the resource allocation for these in-flight instructions in 3D microarchitecture largely affects the reliability of program execution since the stacked dies exhibit heterogeneous soft error resilience. Note that although 3D mitigates the SER of the inner layers due to the shielding effect, the outer die is still as vulnerable as a 2D planar IC. To better exploit 3D’s reliability benefit, dynamically mapping program instructions to 3D microarchitecture across different layers based on their architecture level vulnerability characteristics is proposed. At the architecture level, an instruction can be classified as either ACE or un-ACE. An instruction whose bits are all ACE as is defined as an ACE instruction, otherwise it is referred to as an un-ACE instruction. A method proposed in [37] is used to identify ACE and un-ACE instructions on-line and associate one bit with each instruction to inform its ACE-ness. Alternatively, off-line profiling can be performed to classify ACE and un-ACE instructions. The proposed optimization schemes aim to allocate microarchitecture resources for ACE instructions on layers that are most robust to soft errors.
Figure 2-5 shows reliability-aware resource allocation 3D issue queue (IQ) via instruction scheduling and promotion. It is assumed that a non-compacting issue queue design with an oldest-first selection logic implementation [38] because of its power consumption advantage over the conventional power-hungry compacting approach, and this in turn results in a substantially reduced temperature in 3D designs. Instruction dispatch is modified to perform reliability-aware scheduling. The allocator attempts to find an IQ entry on a robust layer first. As shown by the example in Figure 2-5, instructions ADDQ, STQ and NOP are all allocated on the three free IQ entries on the robust layer (through steps 1, 2, 3). When instructions SUBQ and LDQ are dispatched (steps 4 and 5), the allocator starts the allocation on the vulnerable die since there is no free entry on the robust layers. As can be seen, a naïve reliability-based scheduling scheme can result in scenarios in which the reliable 3D IQ entries are occupied by un-ACE instructions (e.g. NOP in Figure 2-5), while ACE instructions have to be placed on vulnerable dies (e.g. SUBQ and LDQ in Figure 2-5). To address this problem, the allocator is augmented with the capability of making use of the information regarding the occupancy of each die and move ACE instructions across layers by promoting them from vulnerable layers to robust ones when any entry on the robust layers becomes available. Thus, further soft error resilience can be achieved. As shown in Figure 2-5, once the NOP instruction is ready and removed from the IQ (shown as step 6), a promotion takes place to migrate ACE instruction LDQ from the vulnerable layer to the robust layer. To maximize the reliability benefit due to the promotion, each instruction on the vulnerable layer is assigned a priority based on its expected contribution to the vulnerability. Instructions with a higher priority (i.e. the most vulnerable instructions) are promoted and moved to robust layers first. The L2 Cache miss instructions and their dependencies are assigned the highest priority because they experience long delay waiting for the
data returned from memory. Therefore, they expose ACE bits to soft error strikes for a long period of time, leading to high probability of error. These instructions are identified by a waiting bit associated with a physical register and this bit is initially set by a load cache miss. Dependent instructions observe the waiting bit, update its priority value correspondingly, and set the waiting bits of their destination registers. The priority is retained until the load completes. The medium level priority is assigned to ACE instructions that have no ready-operands since the readiness of operands serves as a prediction of waiting time, which in turn determines the exposure time of ACE bits to particle strikes. The operand readiness information can be obtained from a flag bit, which is set upon a match between the broadcasted tag and those of the source operands. Finally, the promotion priority of ACE instructions outweighs un-ACE instructions. The initial priority values assigned at the dispatch stage are either level 2 or 3, since L2 cache misses for load instructions cannot be determined until the later stage. After an instruction is dispatched, its priority value is updated accordingly upon the occurrence of trigging events. As shown in Figure 2-5, the LDQ instruction is a cache miss instruction and gains the opportunity to be promoted through step 7. As a result, LDQ is removed from the queue and reinserted into the free IQ entry on a robust layer (shown as 8).

Allowing instruction migration after the initial allocation increases the number of accesses to the structures due to extra de-queue and en-queue operations. These are likely to result in higher dynamic power consumption caused by the increased switching activity. To minimize the impact of our proposed techniques on a 3D-chip’s temperature, the optimization schemes are designed in a thermal-friendly manner. In the case where several entries become available across multiple dies, the one residing on the die furthest from the heat sink is selected. The reason for that is the promoted instructions are likely to be long-latency instructions such as cache miss
instructions or their dependences. These instructions are likely to occupy microarchitecture entries for a very long time, thereby reducing the dynamic power due to lower activity. This is desirable for layers that have a long thermal dissipation path to the heat sink. The instructions which are nearly ready to be executed are herded towards the top dies to keep the active entries close to the heat sink.

Differing from the IQ, the LSQ and ROB are typically implemented as FIFO queues. Entries are allocated at the fetch or dispatch stage and are deallocated at the commit stage. Take the ROB for an example, entries allocated in the ROB are age-ordered, in other words, the order in which instructions are stored in the ROB match with their order in programs. Instead of scheduling and re-mapping individual instructions, the ROB is split into several logical partitions and a table is used to keep track of the mapping between logical partitions and physical ones on each layer. Figure 2-6 shows an example of the ROB structure. As can be seen, the ROB is partitioned into four segments and each segment is allocated on one layer. A bitmap table is used to map the physical ROB partitions to logical ROB segments. Instead of promoting individual instructions, the content of an entire segment is migrated across layers. To reflect the change of physical location, the bitmap entries for logical segments 1 and 4 are updated so that logical segments 1 and 4 are mapped to the physical segments 4 and 1 respectively. This update ensures correct commitment in the ROB. A similar design applies to LSQ.

**Exploring Narrow-Width Operands and ASRAM for Robust 3D Register File**

Register files hold critical data for correct program execution. An error-flipped intermediate computation result in the register file will likely propagate to later computations, resulting in an erroneous output or crashed execution. However, the register file is in the critical path of the processor pipeline and any reliability optimization that increases the access latency of the register file is less desirable. Therefore, designing both high-performance and error-resilient
register files is critical. Similar to the layout of the ROB, IQ and LSQ described in the previous section, the register file entries can be distributed across layers. The vertical structure of 3D integration offers protection for entries on robust layers against soft errors, while there are still a number of entries exposed to particle strikes on the vulnerable layers. Prior studies have observed that many register values generated and consumed by applications do not require the full width to represent the data [39]. These values are called narrow-width values. Narrow-width values have been previously exploited for improving performance [40] and energy efficiency [41]. Coupled with 3D integration’s shielding capability, this work exploits narrow-width values and reliability-enhanced ASRAM cell to design soft error robust 3D register files with minimized implementation cost. An ASRAM circuit is evaluated from the reliability perspective and it is found that ASRAM is optimal for soft error immunity when storing a preferred value.

Figure 2-7 shows a soft error robust 3D register file design that exploits narrow-width values and ASRAM. As can be seen, each word of a 3D register file entry is distributed to one layer with the 16 Most Significant Bits (MSBs) on die 1 (the layer furthest from the heat sink). Most narrow-width operands contain zeros in their 16 MSBs, while Least Significant Bits (LSBs) do not show any particular inclination to value 0 or 1. In a 3D chip where raw SERs vary across layers, the vulnerability of the MSBs placed on the layers where SER is high can be effectively minimized using ASRAM based cells. The LSBs are assigned to robust layers protected by the 3D chip’s shielding capability. To maximize the protection offered by ASRAM, a supplementary circuit is introduced to store the inverted state “zero” when the MSBs contain all 1s. Each functional unit is augmented with a simple fast leading 0/1 detector to check the state of most significant 16 bits. As computational results are stored in ROB before commitment, an augmented bit, $ROB_{INV}$, is set when 16 MSBs are all 1s. At the commitment stage, these 16
MSBs are converted to 0s before being latched in the register. To indicate the 1-to-0 inversion entries, each register file entry is associated with a bit, $INV$, used to trigger an inversion at the read stage to obtain the correct result.

**Heterogeneous Integration for Cost-effective Reliability Enhancement**

Due to the SER advantage of SOI technologies, stacked dies can be implemented with SOI wafer to reduce the SER. To quantify the impact of SOI on 3D microarchitecture SER, this work conducts circuit level modeling to generalize the physical mechanism of a particle striking SOI devices and investigate the $Q_{critical}$ of circuits using SPICE simulations on a standard 6T SRAM. The experiential results show a 5X reduction in SER by using SOI wafer, which is consistent with previously published experimental results [42]. Unfortunately, from a manufacturing perspective, the primary barrier to widely adopt SOI-based process is the increase of substrate cost. This is because two wafers are required for implementing a SOI wafer. The increased wafer cost contributes a 10-15% increase to the total manufacturing cost [8]. By integrating heterogeneous processing techniques in 3D, it is able to benefit from SOI technologies while minimizing the cost. As shown in Figures 2-3 and 2-4, die 1 is the most vulnerable among all layers and serves as a protecting shield for all other layers. Therefore, instead of applying SOI technologies to all layers, applying SOI to the most vulnerable layer proposed while leveraging 3D shielding effect to achieve our reliability goal while minimizing implementation cost.

**Experimental Methodology**

This section briefly describes the experimental methodology for estimating the SER of the 3D die stacked processor and evaluating performance and thermal impact. Figure 2-8 shows an overview of the SER estimation method. Note that the SERs of 3D ICs are reported as normalized values to that of a baseline 2D planar chip. The normalized raw error rates for each
layer are obtained from the analytical models presented in the previous section. SPICE simulation is used to quantify the soft error rate of the studied radiation hardening circuit (e.g. ASRAM) and manufacturing technique (e.g. SOI) by obtaining the critical charge ($Q_{\text{crit}}$) of SRAM cells. The $Q_{\text{crit}}$ defines the minimal charge deposition required to cause a SEU. It is assumed that a 65 nm technology with a supply voltage of 1.2 volts is used and the model used in the SPICE simulations is the Berkeley 65nm BSIM transistor model [43]. The process of a particle strike is simulated as a current source (Figure 2-9) expressed by the following model:

$$I_{in}(t) = \frac{Q}{\tau_{\alpha} - \tau_{\beta}} (e^{-t/\tau_{\alpha}} - e^{-t/\tau_{\beta}})$$  \hspace{1cm} (2-7)$$

where $\tau_{\alpha}$ is the collection time constant of the junction, and $\tau_{\beta}$ is the ion-track establishment time constant and $Q$ is the charge deposited as a result of particle strike. $Q_{\text{crit}}$ corresponds to the value of $Q$ that produces a large enough current to change the state of the cell. A higher value of $Q_{\text{crit}}$ relates to a more reliable circuit. The results of the SPICE simulations performed are summarized in Table 2-2 to compare the $Q_{\text{crit}}$ and SER of the conventional SRAM to that of the hard-0 ASRAM, assuming the SER of the conventional SRAM is 1.

To analyze performance and reliability characteristics at the architecture level, a reliability-aware cycle-accurate simulator built on top of a heavily modified and extended Simplescalar framework is used. The framework models architecture vulnerability factors of major microarchitecture components (e.g. IQ, ROB, RF, FU LSQ and L1, L2 caches) using methodologies proposed in [44] and [10]. The soft error rate of 3D microarchitecture is derived using its AVF and the raw error rates obtained from aforementioned analytical models and SPICE simulations. The SER of 3D microarchitecture is normalized to the respective 2D planar design. This study models a 3D microprocessor implemented as 4 layers of stacked dies. Table
2-3 summarizes the microarchitecture configuration of the modeled 3D and 2D planar processors. Although the clock frequency of a 3D processor can be improved due to reduced critical path latency [9], it is assumed that the same clock frequency of 3GHz for both cases is used in order to perform a fair comparison of reliability between the two technologies. A collection of integer and floating point benchmarks from the SPEC2000 suite is used in this study. The Simpoint tool is used to choose the most representative simulation interval and a slice of 250M instructions following that point is simulated.

Detailed thermal analysis is conducted to understand the impact of the proposed optimizations on 3D chip temperature. The Wattch power model is integrated into our simulation framework to estimate the power consumption of each microarchitecture structure on each layer. To achieve this, this study keeps track of the energy consumed for each entry of a microarchitecture structure. The thermal analysis tool used is Hotspot [45], which supports thermal modeling of 3D die-stacked processors. The floor-plans used for the baseline 2D processor and each layer of the die-stacked 3D processor are shown in Figure 2-10. The layout of each die in the 3D processor is identical to the 2D planar processor except for a 4x reduction in size due to the partitioned implementation of 3D microarchitectures. Figure 2-3 shows the overall structure of the simulated 3D chip. Aside from the thicknesses information shown on the figure, Table 2-4 lists the dimensional and thermal parameters used in our thermal analysis.

**Results**

This section presents our experimental results comparing the SER of microarchitecture in the 3D chips to their 2D counterparts. Moreover, the additional reliability benefit achieved by the proposed schemes is presented. The performance and thermal impact of our techniques are also included.
**Reliability Improvement due to 3D Shielding Effect**

Figure 2-11 shows the SER reduction of a 3D-implemented processor over the baseline 2D chip. The SER of a processor accounts for major pipeline structures, including IQ, ROB, LSQ, FU and RF. The SER is computed by summarizing the SER for each individual structure. As can be seen, an average of 70% reduction in SER can be achieved by using 3D technology. Note that the SER reduction shown in Figure 2-11 is provided by 3D microarchitecture directly due to the shielding effect of its vertical structure. Whereas, to achieve the same reduction on a 2D design, the conventional reliability optimization techniques require architectural or circuit designs involving a variety of overheads.

Although the SER reduction contributed by 3D technology is substantial, the benefit can be further exploited with the dynamic scheduling and promotion techniques. Figures 2-12 A)-C) present the additional SER reduction achieved by applying the proposed techniques on the IQ, ROB and LSQ structures. On average, a 46%, 30% and 27% of SER reduction is obtained on these three key microarchitecture components respectively compared to the baseline 3D design. This additional reduction is achieved in two ways: First, scheduling attempts to map the vulnerable instructions to robust entries; secondly, promotion allows vulnerable instructions to be relocated to inner layers where the SER is lower. Note that the ROB and LSQ derive less benefit from the proposed optimization technique while larger benefit is obtained on the IQ. This is primarily because the FIFO structure of the ROB and LSQ limits the opportunity of moving in-flight instructions across layers. Even with the segmented design, the relocation of the entire segment to robust layers does not happen very frequently, thereby failing to exploit the reliability benefit offered by die stacking structure.
**Reliable 3D Register File Design based on Narrow-width Operands**

Figure 2-13 shows the SER of the proposed 3D register file optimization that exploits narrow-width operands and ASRAM circuit design. For SPEC integer benchmarks, an extra 13% SER reduction (45% reduction relative to the design of simply distributing register file entries across layers) is achieved when the ASRAM circuit is applied to protect the 16 MSBs against soft error. The primary reason behind this improvement is the operands’ characteristics of integer applications. For SPEC INT2000 benchmarks, around 94% of the integer values can be represented by no more than 34 bits. For SPEC FP 2000 benchmarks, the proposed scheme achieves 9% SER reduction compared with the baseline 3D implementation. The reasons for that are: (1) Memory addresses in the Alpha ISA use 33bits (plus 1 bit = 34bits) and memory operations account for a large portion of the executed instructions. The narrow width operands of these instructions are exploited in floating point applications as well; (2) Integer operations exist in floating point benchmarks and they contribute to the total number of narrow width operands; and (3) Bit 0s in regular values can also be protected by ASRAM. The benefit depends on the number of zero bits present in the MSBs of values. Although floating point applications tends to utilize the entire bit-width, ASRAM’s capability of protecting zero bits covers a large number of cases including narrow-width values.

Two side effects of our proposed register file design are increased write access latency on ASRAM cell and the delay in converting 1s to 0s upon the read accesses. Although ASRAM is desirable for reliability optimization, the write latency is a disadvantage of this design. Increased write access latency affects post-commit latencies such as the time required to copy committed values to the register file entries. The delay in writing back the value into registers may force the instructions waiting for extra cycles to be processed. On the one hand, storing an inverted value of 1s in ASRAM exploits greater reliability benefit; however the cost of an additional cycle of
performing inversion is harmful to performance. To estimate the effect of these extra delays on the performance, the simulator is modified by adding an extra cycle delay to the write latency of register files and one extra cycle to the registers if the inversion bit is set. Figure 2-14 presents the impact of these delays on IPC across all benchmarks. As can be seen, the degradation of IPC is negligible. On average, IPC is degraded by less than 1% across all simulated benchmarks. The reason for such a small degradation is that the 1-to-0 bit flipping at read stage is applied infrequently and the write access delay due to ASRAM doesn’t cause the following instructions to be delayed in execution frequently. Overall these results confirm that the proposed 3D RF design which exploits narrow-width values by using hardened 0-to-1 flip cell is very effective in preventing soft error occurrences in register files while only incurring some minor architectural and manufacturing modifications.

Cost-effective Reliability Optimization by 3D Heterogeneous Integration

Figure 2-15 shows the normalized SER of a 3D processor with heterogeneous integration of SOI techniques. The first four columns present the SER results when only one layer is selected to be manufactured with SOI wafer at a time. The fifth columns show the SER when SOI wafers are used on all layers. Although the lowest SER can be achieved when SOI substrate is applied on the entire chip, the substrate cost associated with this implementation increases by 100%. Whereas, this cost can be reduce to 25% if only a single layer in a 4-layer 3D chip is implemented with SOI substrate. As can be seen from the first four columns, a substantial reliability benefit is obtained when die 1 is selected to use SOI substrate, because it is the most vulnerable layer among all stacked dies. In contrast to the design with SOI substrate applied on die 1, a 4-layer 3D chip with SOI implementation on all layers only reduces SER by additional 6% on average, but results in a 100% increase in the substrate cost. Moreover, with the proposed
technique, an additional average 4% reduction in SER can be achieved on top of the 3D design with SOI implementation on die 1, as can be seen by the last column in the Figure.

**Thermal Impact**

Thermal issues are a major concern for 3D chip design. The proposed scheduling and promotion technique requires extra accesses to physical entries for relocating instructions, thereby increasing the dynamic power consumption. Thermal profiling is conductor to investigate the effect of the scheme on the 3D chip’s temperature. Figure 2-16 shows the thermal characteristics of each layer when running the benchmark *bzip* which induces the largest power consumption overhead by our proposed schemes. The first column shows the average temperature of each microarchitectural component residing on a layer of the 3D processor without reliability optimization technique enabled, while the second column shows the temperature when optimization techniques are enabled. As can be seen from the figure, there is a slight drop in the temperature, particularly for the structures residing on die 1 (furthest from the heat sink). This drop in temperature can be explained by the fact that our allocation algorithm is designed to be thermal-friendly. The promoted instructions are relocated to inner layers which are second furthest from heat sink because those instructions are likely to be long-latency ones such as cache misses instructions and their dependence. While instructions which are nearly ready to execute are herded towards the top dies to keep the active entries close to the heat sink.

**Related Work**

Many recent research efforts have explored the performance benefits and thermal issues of 3D integration technique. [2][3] started the investigation of the performance impact of placing main memory or cache on top of the processor to form a 3D chip. Some other studies focused on splitting individual unit blocks and replacing long-delay wires with shorten vertical interconnects for a true 3D design with reduced operating latency and power consumption, such as thermal and
performance analysis [5], arithmetic units design [46], register file implementation [47] and stacked DRAM [1]. [48] proposes implementing a redundant checker processor on a second die to verify the execution results by using 3D fabrication technologies. The primary difference between this work and theirs is that we investigate the heterogeneous SERs of vertically stacked dies from the soft error physical mechanisms and further exploit it using architecture level optimization schemes, while [48] uses 3D technique as an enabler to facilitate redundant execution. In addition, the 3D structure studied in this paper is a true 3D design which is more promising, while [48] does not fully exploit 3D stacking technology because the individual structures are still inherently two dimensional.

Various microarchitecture soft error mitigation techniques have been proposed over the last few years. [37] proposes an IQ design consisting of standard SRAM cells and radiation hardened rSRAM to enhance the soft error robustness. [49] uses ASRAM circuit design optimized for reliability hardening to store zeros in FPGA for reducing the failures. [50] propose a novel framework based on redundancy addition and removal (RAR) for soft error rate (SER) reduction in combinational logic. Several studies proposed to mitigate soft error vulnerability at architecture level, such as squashing long-delay instructions [51] and using redundant execution to verify the correctness of program execution [52][53][54]. The most relevant study on narrow-width value based reliability optimization is [55], in which the narrow width values were exploited for improving register file reliability by duplicating value inside a single entry. However, the access latency associated with this technique is not desirable and the study focuses on 2D planar design. In this work, the reliability strength of ASRAM design at the circuit level is combined with narrow width value characteristics at architecture level to reduce SER as well as minimizing the performance penalty introduced. More importantly, this work focuses on 3D
design and take advantage of the shielding effect offered by stacked dies to deploy reliability-hardening techniques selectively to the most vulnerable layers so that the complexity and cost involved at the circuit design and manufacturing stages can be minimized.
Table 2-1. A Summary of parameters in modeling a striking of alpha particle on silicon

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>Energy of incident particles</td>
<td>Initial energy range 4MeV ~ 9MeV</td>
</tr>
<tr>
<td>K</td>
<td>Factor = (4\pi N_c r_e^2 m_e c^2)</td>
<td>0.307 MeV cm(^2) mol(^{-1})</td>
</tr>
<tr>
<td>r_e</td>
<td>Classical electron radius</td>
<td>2.817 \times 10^{-13} \text{cm}</td>
</tr>
<tr>
<td>m_e</td>
<td>Electron mass</td>
<td>9.11 \times 10^{-31} \text{kg}</td>
</tr>
<tr>
<td>N_a</td>
<td>Avogadro’s number</td>
<td>6.022 \times 10^{23} \text{mol(^{-1})}</td>
</tr>
<tr>
<td>z</td>
<td>(\varepsilon e) charge of incident particle</td>
<td>+2e, where e = 1.602 \times 10^{-19} \text{C}</td>
</tr>
<tr>
<td>A</td>
<td>Atomic number of absorber</td>
<td>14 g \cdot \text{mol}^{-1}</td>
</tr>
<tr>
<td>\beta</td>
<td>Velocity of particle over the speed of light</td>
<td>(v/c)</td>
</tr>
<tr>
<td>\gamma</td>
<td>(1/\sqrt{1-\beta^2})</td>
<td>0.510998918 MeV</td>
</tr>
<tr>
<td>m_e c^2</td>
<td>Electron mass \times c^2</td>
<td></td>
</tr>
<tr>
<td>T_{max}</td>
<td>Maximum kinetic energy which can be imparted to a free electron in a single collision</td>
<td>for higher energies</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for lower energies</td>
</tr>
<tr>
<td>\rho</td>
<td>Density of silicon</td>
<td>2.33 g \cdot \text{cm}^{-3}</td>
</tr>
<tr>
<td>I</td>
<td>Mean excitation energy</td>
<td>175 eV</td>
</tr>
</tbody>
</table>

Table 2-2. Comparison of Qcrit and SERs for ASRAM and SRAM

<table>
<thead>
<tr>
<th>Parameter</th>
<th>SRAM Qcrit /SER</th>
<th>ASRAM Qcrit /SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>Storing 0</td>
<td>58fc / 1</td>
<td>92fc / 0.64</td>
</tr>
<tr>
<td>Storing 1</td>
<td>22fc / 1</td>
<td>20fc / 1.20</td>
</tr>
</tbody>
</table>

Table 2-3. Base configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>4-wide fetch/issue/commit</td>
</tr>
<tr>
<td>Issue Queue</td>
<td>96</td>
</tr>
<tr>
<td>ITLB</td>
<td>128 entries, 4-way, 200 cycle miss</td>
</tr>
<tr>
<td>Branch Predictor</td>
<td>2K entries Gshare, 10-bit global history</td>
</tr>
<tr>
<td>BTB</td>
<td>2K entries, 4-way</td>
</tr>
<tr>
<td>Return Addr Stack</td>
<td>32 entries RAS</td>
</tr>
<tr>
<td>L1 Instruction Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle access</td>
</tr>
<tr>
<td>ROB Size</td>
<td>128 entries</td>
</tr>
<tr>
<td>LSQ</td>
<td>48 entries</td>
</tr>
<tr>
<td>Integer ALU</td>
<td>4 I-ALU, 2 I-MUL/DIV, 2 Load/Store</td>
</tr>
<tr>
<td>FP ALU</td>
<td>2 FP-ALU, 2 FP-MUL/DIV/SQRT</td>
</tr>
<tr>
<td>DTLB</td>
<td>256 entries, 4-way, 200 cycle miss</td>
</tr>
<tr>
<td>L1 Data Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle access</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>shared 4MB, 8-way, 128 Byte/line, 12 cycle access</td>
</tr>
<tr>
<td>Memory Access</td>
<td>64 bit wide, 400 cycles access latency</td>
</tr>
</tbody>
</table>
Table 2-4. Thermal parameters used in thermal analysis

<table>
<thead>
<tr>
<th>Type</th>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dimensional Parameters</td>
<td>Thermal Interface Material (TIM)</td>
<td>50um in thickness</td>
</tr>
<tr>
<td></td>
<td>Heat Spreader</td>
<td>3cm X 3cm X 1mm</td>
</tr>
<tr>
<td></td>
<td>Heat Sink</td>
<td>10cm X 10cm X 2 cm</td>
</tr>
<tr>
<td>Thermal Parameters</td>
<td>Silicon Thermal Resist</td>
<td>0.0083 W/mK</td>
</tr>
<tr>
<td></td>
<td>Thermal Interface Resist</td>
<td>0.5000 W/mK</td>
</tr>
<tr>
<td></td>
<td>Heat Sink Resist</td>
<td>0.1000 W/mK</td>
</tr>
<tr>
<td></td>
<td>Ambient Temperature</td>
<td>310K</td>
</tr>
</tbody>
</table>
Figure 2-1. Energy loss of alpha particle in silicon.

Figure 2-2. Possession energy of alpha particle during striking process

Figure 2-3. A cross-section view of the simulated 3D processor.
Figure 2-4. Energy spectrum of alpha particle and the % of particles reaching active layers on each die.

Figure 2-5. Reliability-aware instruction scheduling and dynamic promotion in 3D issue queue.

Figure 2-6. Segmented 3D ROB design for reliability-aware scheduling and promotion.
Figure 2-7. Narrow width value and ASRAM based 3D register file design.

Figure 2-8. An overview of 3D die stacked processor SER estimation.

Figure 2-9. Circuit simulation of particle strike on SRAM.

Figure 2-10. The floor-plans of simulated 2D and 3D processors.
Figure 2-11. Normalized SER of 3D implementation.

Figure 2-12. A) The IQ reliability improvement achieved by scheduling and promotion technique. B) The ROB reliability improvement achieved by scheduling and promotion technique. C) The LSQ reliability improvement achieved by scheduling and promotion technique.
Figure 2-13. The impact of reliable register file design on SER.

Figure 2-14. The impact of reliable register file design on IPC.

Figure 2-15. SER comparison of different SOI integration schemes under 3D technologies.

Figure 2-16. Chip temperature of (1) the baseline 3D microarchitecture design and (2) 3D microarchitecture with reliability optimizations.
In contrast to conventional memory technology, a cutting-edge memory technology, Phase-change Random Access Memory (PCM), is attracting increasing attention as a promising candidate for next generation memories [16][17][18][19][56]. PCM is a type of non-volatile memory that uses the unique behavior of chalcogenide glass, which can be switched between two states (i.e. crystalline and amorphous) with the application of heat. The desirable characteristics of PCM include random access, fast read access, low standby power, superior scalability (no physical limits down to 20nm technology node [20]), compatible with CMOS process[21] etc. This section presents a case in which PCM, as a replacement for DRAM, can be employed to implement 3D-stacked memory systems with lower power consumption and alleviated temperature constraints. The proposed approach leverages two attractive features provided by PCM: low standby power and high-temperature friendly operation. The former is a common feature of all non-volatile storages as data can be retained in them even when not powered. The latter is a unique characteristic of PCM: to store data in PCM, the temperature needs to be elevated to switch the state of cells. Using analytical and circuit-level modeling that characterizes PCM in detail, it is observed that the programming power of PCM cells can be reduced as the chip temperature is elevated, in contrast to the exponential increase in refresh power for DRAM when chip temperature increases. This high-temperature friendly feature makes PCM superior to DRAM for die-stacked memory systems. In addition, the power benefit of using 3D TSVs to deliver PCM programming current is investigated. Compared with conventional metal wires, the low resistance of TSVs minimizes the dissipated power along the bit lines. It is shown that PCM benefits more from TSVs than DRAM in terms of power savings.
Two major challenges that need to be addressed for design using this emerging memory technology are PCM high write latency and limited endurance. In this research, a hybrid main memory design is proposed which is composed of a large portion of PCM used as a primary memory space and a small portion of DRAM that serves as a write buffer to reduce the number of writes to the PCM partition. To maximize the runtime efficiency of the DRAM-based write buffer, it is proposed that an OS-level paging scheme that takes into account the memory reference characteristics of applications and migrates the hot-modified pages from PCM to DRAM so that the life time degradation of PCM is alleviated. This hybrid design with page migration also provides an improved memory write performance over PCM-only memory design due to the lower write latency on DRAM than that on PCM. As writes to the PCM cannot be entirely avoided and the wear-out on a given PCM cell is exacerbated as the number of overwrites increases, it is crucial to provide an endurance optimization scheme that can maximize and balance the life span of all PCM cells. Toward this end, a synergetic reliability enhancement approach is proposed that combines architecture- and OS-level wear-out optimizations. At the architecture level, the proposed approach uses Error Correction Code (ECC) with varied strength to detect and correct a number of errors in each memory block fetched. However, the performance overhead (increases with the number of errors that need to be corrected due to the enlarged delay in error correction) of using ECC with increased strength is non-trivial. To minimize the ECC-induced error correction penalty in performance and achieve wear-leveling, it is proposed to use an OS paging scheme to perform endurance-aware allocation.

**PCM Power Characterization under 3D Integration Technology**

By stacking memory directly on top of the microprocessor, [5] reports a 65% performance gain by placing a planar DRAM on top of the processor and [1] further shows an 175% speedup over die-stacked planar DRAM by proposing a true-3D design. However, the increased
temperature of 3D chips causes the DRAM to consume more refresh power [6]. The heat-driven 
programming mechanism of PCM is more temperature-friendly than DRAM, motivating us to 
explore PCM as an alternative candidate for die-stacked memory design. This research for the 
first time presents a detailed characterization of the impact of high temperature on PCM 
programming power and further investigates the benefit of 3D TSVs on PCM programming 
power. The results of this work show that PCM-technology is well suited for 3D-stacked 
memory implementation.

**Impact of 3D High-Temperature on PCM power**

PCM has substantially reduced read and standby power consumption. As a result, overall 
PCM power is dominated by its programming power (i.e. SET and RESET power). To 
characterize the impact of temperature on the PCM programming power, the one-dimensional 
heat conduction model [57] is used. This heat conduction model captures the flow of heat in 
PCM device that is insulated everywhere except at the two ends, which connect to bit-line and 
access transistor. The elevated ambient temperature can reach the PCM devices and help heat 
them through the two ends. On the other hand, the generated thermal in PCM devices may also 
dissipate to the surrounding area, increasing on-chip temperature. Overall, PCM programming 
activity has little impact on chip temperature due to the small amount of concurrent writes at 
anytime. In this section, only the analysis for the RESET operation is presented briefly (refer to 
[57] for details) and a similar characterization procedure has been applied to the SET operation.

The overall structure of the modeled PCM device is shown in Figure 3-4. The same 
dimensional parameters as those in [57] are used. Phase switching of Ge₂Sb₂Te₅ is triggered by a 
Joule resistive heater \( q_{\text{joule}} \), which can be specified as following

\[
q_{\text{joule}} = I_{\text{j}}^2 R t
\]  
(Eq 3-1)
where $I_F$ is the current passing through the device, $R$ is the resistance of the Joule heating material, and $t$ is the heating time. The heat conduction model (Eq 3-2) is used to capture the temperature profile shown in Figure 3-4.

$$
p_i \frac{\partial T_i}{\partial t} = \frac{\kappa_i}{c_i} \frac{\partial^2 T_i}{\partial x^2} + \frac{q_{Joule,i}}{c_i}
$$

(Eq 3-2)

where $x$ is the distance from the active region of Si substrate, $T_i$ is the temperature, $t$ is the heating time, $\rho_i$ is the density, $\kappa_i$ is the thermal conductivity, and $c_i$ is the specific heat capacity for the $i$th layer (i.e. Bottom Electrode Layer, Heater Layer and GST Layer) with $i=1,2,3$. As described in the previous section, a temperature rise due to the heat generated by the reset current is required to melt a small region of phase change material adjacent to the heater to achieve a successful RESET operation. The minimal thickness of this required region is defined as $Thick_{crit}$ (as illustrated in Figure 3-4), which will determine the minimal required programming current. By using Eq 3-1 and Eq 3-2, the minimal required programming power is computed as the ambient temperature varies from 40°C to 95°C, a range that covers the temperature of both planar and 3D die-stacked chips [1].

Figure 3-5 illustrates the effect of chip temperature on a PCM device’s programming power: less programming current is required to RESET/SET a PCM device at elevated temperature, resulting in smaller programming power. The reduction for RESET and SET power is 15.7% and 19.4% respectively when the ambient temperature increases from 45°C to 85°C. Note that the temperature threshold for RESET and SET is ~600°C and ~300°C respectively. A 19.4% power saving is achieved for SET when the temperature gap is bridged by 13% (40°C/300°C=13.3%). A lower percentage of power saving (i.e. 15.7%) is obtained for RESET due to a smaller gap (i.e. 40°C/600°C=6.7%) bridged. The power saving percentage is greater
than that of temperature increment. This is because the flow rate of heat dissipated from the PCM device through the two ends is proportional to the temperature gradient between the device and surrounding materials. In addition to bridging the temperature gap, the elevated on-chip temperature reduces this gradient and increases heating efficiency, resulting in further power saving. The results are largely consistent with [58], which reports experimental data on temperature dependence of RESET power and shows a 25% power saving when temperature is elevated by 16% ($100^\circ$C/$600^\circ$C=16%). In contrast to PCM, DRAM consumes more power at an elevated temperature due to the increased refresh frequency (driven by temperature rise). A typical DRAM refresh interval is 64ms for 2D planar chips [6][59]. As 3D-stacked on-chip DRAM is employed, the refresh rate is required to be at least doubled if the operating temperature exceeds $85^\circ$C [59]. The power models from DRAMsim [60] is used to estimate the power overhead of shortening the refresh interval from 64ms to 32ms when running simulated workloads. The results show an average 3X increase in refresh power and an average 8% increase in total power for die-stacked DRAM. As the ambient temperature increases in 3D stacking design, thermal cross talking may cause PCM programming currents for one cell to interfere the states of adjacent cells. Thermal cross-talk effect for PCM devices at 65nm technology is simulated and the results show the temperature falls exponentially with the increased distance from the programmed cell. This temperature becomes close to the ambient temperature at the adjacent cells, suggesting no thermal coupling effect on 3D chip with elevated temperature.

**Impact of 3D TSVs on PCM Power**

Through Silicon Vias are ubiquitously adopted in 3D design to reduce wire-length, thereby achieving power reduction. This work analyzes the power benefit obtained by TSVs in 3D-
implemented DRAM and PCM and show that TSVs benefit PCM power savings more significantly than they do for DRAM.

The dynamic power \( P_{\text{dyn}} = fCV_c^2 \) due to charging and discharging interconnect capacitance is one of the primary contributors to the power consumption of read and write operations for DRAM cells. By using TSVs, the substantial wire length reduction decreases the interconnect capacitance, leading to less dynamic power consumption. Intuitively, since \( C \) is proportional to the wire length, given a fixed \( f \) and \( V_c \), the dynamic power reduction is proportional to the wire length reduction. However, the dynamic power saving of true-3D DRAM is less than the fraction of wire length reduction (detailed later in this section). This is because the TSVs used to form the vertical bus are of high capacitance [5], which offsets the interconnect capacitance reduction due to the shortened interconnect, leading to a substantial loss in power savings. Differing from DRAM, the power dissipation of write operations for PCM is dominated by the programming power \( P_{\text{programming}} = I^2R \) rather than \( P_{\text{dyn}} = fCV_c^2 \). The reasons are two-fold: first, PCM write operations are carried out only on those PCM cells that are selected by both row decoder and column decoder. Therefore, only bit lines that connect to those selected PCM cells will be charged or discharged during write operations, leading to a substantial reduction in \( C \) as well as \( P_{\text{dyn}} \). In contrast, the read/write operations in DRAM will cause all cells in a selected row to be read out / written back, because capacitors in this row are connected to bit-lines once the row is activated. These capacitors then start to charge or discharge all bit-lines, resulting in a significantly higher \( C \) in \( P_{\text{dyn}} = fCV_c^2 \) for DRAM. Second, the magnitude of the PCM write current required to flow through the direct current path between VDD and GND during write operations is very high. Therefore, the power dissipated along the path due to the resistance outweighs the power required to charge interconnect wires. By using
TSVs in die-stacked PCM design, the resistance is decreased due to the shortened interconnect, thereby saving programming power. More importantly, the physical dimension of TSVs makes their resistance per unit length much smaller. Hence adopting TSVs in die-stacked PCM retains the power savings benefit due to the reduced resistance (as a result of short wires) along the current path. More interestingly, it is observed that the reduced resistance along the current path allows the programming current to be further reduced, while still being able to achieve successful programming operations. Since a smaller magnitude of current is used, power saving can be achieved.

To characterize PCM power under 3D TSVs, a 256Mb PCM similar to the prototype developed by [60] (the layout is shown in Figure 3-6) is modeled. Instead of using the 2D planar architecture and layout, the true-3D [1] structure announced by Tezzaron Corporation is adopted, which constructs on-chip DRAM memory by stacking individual bit-cell arrays in a 3D fashion [13]. Figure 3-7 shows an overall organization of a true-3D PCM. The bit-cells belonging to one bank are distributed across the top 4 layers and interconnected by TSVs. A dotted line in the planner design represents a row of memory cells. In a true-3D design, the row is divided into four sections with each section residing on one layer, as is shown by four dotted lines at the top four layers in true-3D organization. The peripheral circuits are located at the bottom layer for speed optimization. Circuit model is build and SPICE simulations are performed to quantify the overall benefit of TSVs on PCM programming power. The SPICE results show that PCM programming power reduces from 1.2mW (planar design) to 0.81mW (3D design), a 32.5% power saving. Additional circuit models are built for DRAM and SPICE simulations are performed to evaluate the dynamic power savings achieved by a 4-layer true-3D DRAM. The results show that the percentage of dynamic power savings is only 20%, considerably less than
that achieved on true-3D PCM. By combining the 32.5% power savings due to TSVs and 19.4% power savings due to elevated temperature, it is estimated that up to a 46% PCM power savings can be achieved by 3D.

**The Proposed 3D Die-Stacked Hybrid PCM/DRAM System**

The low standby power and temperature-driven operation of PCM technology make it well suited for die-stacked memory systems from the perspective of power savings. However, there are several challenges (i.e. write bandwidth, endurance) that need to be addressed for the practical and effective integration of die-stacked PCM. This work proposes a hybrid, true-3D-implemented PCM/DRAM memory architecture along with architecture and OS support to address these challenges.

**An Overview of the Hybrid Memory**

Figure 3-8 A) shows an overview of the proposed PCM-based 3D memory architecture. The proposed design consists of a true-3D-implemented 2048MB PCM and a planar 128MB DRAM partition to exploit the heterogeneous performance and power characteristics of the two. The high-speed, power-efficient write access and the infinite lifespan of DRAM make it suitable to serve as a write partition since doing so improves memory write bandwidth and avoids PCM’s expensive write power and wear-out. On the other hand, the very low power of PCM allows greater power savings. OS-level paging is used to migrate pages to the DRAM partition based on a program’s run-time memory reference characteristics.

As the chip temperature affects PCM programming current, the memory controller is augmented with the capability of adapting the current used for programming PCM cells based on the ambient temperature. 16 digital thermal sensors similar to those used by Intel Core Duo [61] are employed on PCM layers with 4 sensors per layer to collect run-time temperature profile and feed it back to the memory controller. The sensors are placed in locations where hotspot is likely
to appear. With the collected run-time temperature, memory controller sends signals to PCM peripheral circuits to indicate the current level used for memory writes. The write driver circuit block (shown in Figure 3-3) is augmented to implement a current-adjustment scheme with 8-level tuning resolution. This tuning resolution is determined by making a tradeoff between design complexity/overhead and performance. The estimated area overhead of this scheme is 0.02 mm². The memory controller and memory modules are connected by 3D TSVs, which form a fast, vertical interconnect across layers. Note that the width of this TSV-based bus is not limited by the pin-count of the off-chip memory. A bus with a width of 73 Bytes (i.e. 64 Byte data + 9 Byte ECC code) is assumed and the type of ECC code adopted is a BCH code to ensure the reliability of the memory system and improve its lifetime by varying its error correction capability. The ECC encoding and decoding (including correcting errors if necessary) are performed on each memory access. This ECC functional block is implemented using dedicated hardware. The number of corrected errors by the ECC hardware is passed to the OS (through an interrupt when a correction occurs) to adapt its page allocation scheme by favoring lightly used physical pages over heavily used ones during memory allocation. This achieves a balanced wear-out on the PCM partition. It also offers a performance benefit, because accessing heavily worn-out physical memory incurs considerable delay in ECC decoding due to the time required for correcting errors.

As shown in Figure 3-8 B), a quad-core microprocessor with a shared L2 cache with die-stacked on-chip memory is modeled and each core is an Alpha 21364 microprocessor core. They are organized as a six-layer 3D-stacked chip by allocating the quad-core on Layer_0 (i.e. the layer that is closest to the heat sink for the purpose of thermal efficiency). Layer_1 is used for planar 128MB DRAM, the memory controller, PCM memory peripheral circuits, and ECC
encoder and decoder with optimized speed for CMOS technology. Layer_2 to Layer_5 are dedicated for PCM bit-cells (consisting of a PCM device and an access transistor), which are designed in a true 3D fashion and realized based on a traditional NMOS technology optimized for density. Note that in the proposed design, DRAM and PCM are in different layers to reduce fabrication cost since the PCM fabrication process is different than that of the DRAM. The DRAM partition consists of one bank and the PCM partition has eight banks.

PCM-aware OS Paging

Upon a paging request, a conventional OS virtual memory management scheme allocates the next available page(s) without taking into account the characteristics of the underlying storage media. The key idea of our PCM-aware paging scheme is to favor PCM over DRAM when allocating cold-modified pages which are infrequently updated, so that the very low standby and read power benefits of PCM can be fully exploited, while allocating hot-modified pages to the DRAM partition to avoid the write latency and mitigate wear-out of PCM.

First, it is assumed that pages used for kernel space are always located on the DRAM partition, as they are very likely to be modified intensively. For pages used for user space, counters are used to track the frequency of page updates and adopt a modified Multi Queue Algorithm [62] to classify hot- or cold-modified pages. Specifically, multiple LRU queues (denoted as Q₀, Q₁ … Qₙ₋₁) are used with a different rank for each queue. When a page is modified for the first time, its page number is inserted into the tail entry of the queue with the lowest rank (i.e. Q₀) and the modification counter associated with this page is set to 1. Later, if the same page is updated again, its modification counter is updated and the page number is removed from its current LRU queue Qᵢ and is placed to the tail entry at another queue with an m-higher rank, Qₖ, where m is a function of its modification counter f. m = \log₂(f) is used in our design as a prior study [62] shows this function outperforms others. A periodic demotion of all
page numbers in the queues is performed by degrading page numbers from their current queue to
a one-lower ranked queue and by discarding those page numbers stored in the lowest-ranked
queue. In addition, all modification counters associated with page numbers are halved by shifting
right one bit. In the normal Linux OS, any context switch interval between 10−200 ms may be
used. 10ms is used as the periodic demotion interval to make the page classification aware of the
altered memory reference behavior due to running different programs caused by a context
switch. 16 queues are used and pages are categorized in the 8-highest ranked queues as hot-
modified pages, while the pages in the remaining 8 queues are regarded as potential hot-modified
pages. Previous research has shown that as few as 8 queues can be sufficient to separate hot
pages from others [62].

For the initial page allocation, the memory requests are satisfied by allocating physical
pages in PCM. Pages in the 8-highest ranked queues are migrated to DRAM and set a bit
associated with its entry indicating the location of these pages. If there is not enough space in the
DRAM, the pages whose page numbers don’t exist in the 8-highest ranked queues will be
migrated from DRAM back to PCM (or written back to disk if no space available in PCM). Then
its page number will be removed from the queue. This prevents the frequent migration back and
forth between DRAM and PCM, since pages evicted from DRAM need to be updated frequently
before they are promoted to the 8-highest ranked queues. Page migrations are performed
transparently to the program with the aid of the OS, which is responsible for maintaining TLB
coherence, copying the page to its new home (we emulate this migration in our simulations by
invoking a bcopy() routine) and flashing the cache lines belonging to the pages to be migrated.
Due to the die stacking, the high bandwidth between cores and memories allows migrations to be
accomplished with significantly lowered latency as compared to a conventional off-chip memory
organization. In this case, each queue contains 4,096 entries so that a total of 65,536 page numbers can be stored in 16 queues (each queue entry is 8Byte, among which 4Byte is used for page number and 4Byte is allocated for counter), implemented using a 512KB DRAM with an estimated die area overhead of 0.24mm². Note that an entry is not allocated for every page. An entry is allocated only when the page is recently modified. The periodic demotion will deallocate entries if their associated pages are not updated frequently enough. The 32,768 page numbers stored in the 8-highest ranked queues correspond to a size of 128MB memory area, which is size of DRAM partition in our hybrid design. The counters are incremented through read-modify-write operations in parallel with memory accesses to avoid performance overhead. This multi-queue algorithm is implemented in memory controller with an area overhead of 0.11mm². It is estimated that the power overhead as approximately 11.1mW by using circuit-level tools and modified DRAMsim simulator.

**Life Span Optimization using Varying ECC Strength and OS-level Wear Leveling**

As write operations to PCM are unavoidable, using advanced error correction code (ECC) with OS-level scheme is proposed to ensure the reliability and to tolerate multi-bit failure in data blocks for expanding the life span of PCM with minimum performance overhead. The most commonly used error correction code for DRAM is Hamming code, which can only correct a single-bit error in a message. To increase PCM life span, an advanced ECC, Hamming and Bose-Chaudhuri-Hocquenghem (BCH) code [63], is used to correct up to 7 errors. Due to the read/write pattern and endurance variation (e.g. caused by process variation), a fraction of cells may fail far earlier than others (i.e. failures occur at the early age of the product). When memory references occur on these cells, ECC will inevitably harm the system performance due to the delay in correcting errors. Therefore, feeding the number of errors corrected to OS for achieving wear-out aware page allocation is proposed. This augmented page allocation scheme favors
lightly worn pages over heavily worn ones. Doing so can achieve wear-leveling as well as performance benefit by minimizing performance overhead of BCH.

For each message data of $k$ bits, a BCH codeword (message data + extra redundancy bits) is constructed with a length of $n$ bits to correct up to $t$ errors out of the entire codeword. The length of the codeword, $n$, should satisfy $n=2^m-1$ and $mt \leq n-k$. In this study, an on-chip bus with a width of 64 Bytes is assumed. Due to storage overhead, the number of correctable errors is limited to 7 per BCH codeword. Given a value of 512 bits (64 Byte) for $k$ and 7 for $t$, $m$ can be found to be 10, and at least 70 ($m \times t = 70$) check bits are required. 9 Byte (72 bits > 70 bits) are used as redundancy bits for each 64 Byte data. The area overhead is 14.1% (i.e., 9Byte/64Byte=0.141), which is slightly higher than the overhead (8bit/64bit = 0.125) of using Hamming-based ECC. Note that BCH decoding (encoding) occurs on every PCM read (write) access. The decoding operation also includes error correction if errors are detected. The hardware-based encoder and decoder design proposed in [64] are adopted and it is estimate that the die area required for this hardware implementation is 1mm². As shown in Figure 3-9, the performance overhead of BCH decoder rises substantially (i.e. from 3ns to 26ns) as the number of errors increases from 1 to 7. The estimated maximum power consumption due to tracking multiple bits in ECC is 0.2W, which occurs when 7 errors are required to be corrected.

To achieve endurance-aware OS page allocation, the default page allocation algorithm is modified to take wear-out level into consideration. Linux uses three buddy systems to handle page frames in DMA, normal and high memory zones respectively. The kernel delays allocating dynamic memory to user processes until page fault exception occurs. All free page frames are grouped into 11 lists of blocks that contain a variable number of contiguous page frames, as shown in Figure 3-10. Each block consists of $2^k$ pages in the $k_{th}$ list and the descriptor of the first
page of a block is used to represent this block in the link-list. Blocks within a given list are unordered, so the allocation and de-allocation of pages are performed without considering the wear-out level of physical memory cells. It is proposed to augment the buddy allocator with wear-out awareness. The key idea is that pages with a lower level of wear-out are allocated before others. As shown in Figure 3-10, the single list is transformed into multiple lists, one for each wear-out level (e.g. eight lists for maximum seven correctable errors in our case). The wear-out level, which is indicated by the maximum number of errors detected and corrected in all memory references to a physical memory page, ranges from 0 to 7. With the proposed scheme, pages are placed into multiple lists according to their wear-out level and they are unordered in these lists. Therefore, page allocation and de-allocation from a given list still remain an O(1) operation. Our proposed scheme requires a modification of existing page management routines and data structures. A variable called \texttt{page\_wear\_out} is added to the page descriptor to track the wear-out level of a physical page. The value of \texttt{page\_wear\_out} is updated using the feedback from ECC hardware. The initial value of \texttt{page\_wear\_out} for each page is 0 and it increases as more errors are detected and corrected. A page allocation request is satisfied from the list with the lowest wear-out level first. If there are no free pages available, other lists of higher wear-out levels are considered in-order. For blocks with more than one page, another added variable called \texttt{block\_wear\_out} is used, which represents the average wear-out level of a block, and store the information in the page descriptor of the first page of the entire block. The value of \texttt{block\_wear\_out} needs to be updated during allocation and de-allocation, such as 1) splitting a block with a larger number of pages than the requested memory size in half; 2) merging together pairs of free blocks into a single block with doubled size. In addition, \texttt{block\_wear\_out} is updated when the \texttt{page\_wear\_out} of any page within this block changes. With
the proposed approach, pages with lower wear-out level are favored over others and the performance overhead of BCH code is minimized since fewer errors need to be corrected for lightly worn-out regions than for heavily worn-out ones, since less errors occurs in memory reference to the former, thereby less time required in correcting errors. The proposed scheme conservatively categorize a page as heavily worn even if a few cells in a row fail early, because it is possible that more writes will be performed on this row and cause more cells start to fail. Eventually ECC is not capable of correcting all errors. In this case, the entire page cannot be used since OS manages memory at page granularity.

Note that the proposed technique is different than that used by flash, which relies on the Flash Translation Layer implemented at the file-system level, thereby involving significant performance penalty and complexity. The proposed scheme is built on exiting OS-page allocation algorithms to achieve wear-leveling with small overhead. Since the wear-out information is stored in memory as run-time data, it may be lost due to power off, OS reboot and reinstallation etc. By feeding back the number of corrected errors from hardware ECC block to OS, the proposed scheme allows the OS to regain physical memory wear-out statistics through a short period of learning process.

In addition to page allocation, the page replacement policy by using a second-chance algorithm is also augmented to take wear-out into consideration. More specifically, the default OS page reclaiming algorithm is used to select top ten pages that are ready to be replaced. By examining the wear-out level of these pages, the one that with the lowest wear-out level is chosen to be freed or written back to disk if it is dirty. This allows wear-out awareness to be considered even when there isn’t plenty of free memory frames available, which may occurs frequently in server system.
Experimental Methodology

A quad-core processor with a shared 2M cache is simulated. It is assumed 3GHz frequency and a 65nm technology with a supply voltage of 1.2V. Table 3-1 summarizes the baseline machine architecture. The proposed techniques are evaluated by using both SPEC multi-programming workloads and memory-intensive multi-programming workloads shown in Table 3-2. For SPEC multi-programming workloads, benchmarks from SPEC2000 with reference inputs are selected. For each benchmark, Simpoint[65] is used to find representative simulation intervals and select the interval with the highest MPKI. To form multi-programmed workloads, benchmarks are firstly categorized into: high-miss (MPKI≥20), moderate-miss (5<MPKI≤20) and low-miss (MPKI≤5) categories. In Table 3-2, the High-, Moderate- and Low- miss workloads (H1-H4, M1-M4, L1-L4) consist of four benchmarks exclusively from each category. The High-Moderate- and Moderate-Low- miss workloads (HM1-HM4, ML1-ML4) are formed by using two benchmarks from each category. To stress the memory system intensively, a diverse set of memory-intensive applications is used from various suites which feature gigabyte working sets: Triad (a streaming benchmark derived from the STREAM suite), Qsort (a Unix utility), Kmean from MineBench suite and six applications (BT, CG, FT, LU, MG and SP) from NAS Parallel Benchmarks version 3.2 with class “C” input sets. Four copies of each application are used to form 4-threaded memory-intensive multiprogramming workloads in Table 4-2.

To investigate the performance and power impact of the PCM-based memory architecture, a framework based on a heavily extended Sim-Alpha Simulator [66] is developed with integrated modified memory model, DRAMsim [60], and the Wattch Power Model [67]. DRAMsim is used for both timing analysis and memory power evaluation, while the power consumption of the cores and caches is calculated by Wattch Power Model. The leakage power of processor cores and caches is assumed to be 15% of their dynamic power based on data reported in [68]. The
power overheads of advanced ECC scheme and 16 queues used for OS-level paging are also included in our power estimation. Given the limitation of the simulation infrastructure, memory-intensive workloads are simulated by using memory traces generated from PTLSim/X full system simulator and feeding them into the modified DRAMsim to measure memory performance and power. It is assumed that disk access latency is 4.2ms [63] on IDE. Conventional DDR2 memory is used and the configuration is shown in Table 3-3. For the PCM-based memory, simulations are performed using two sets of parameter values: average case and aggressive case. For the average case, the range of values for a given PCM parameter is found through an extensive literature search and use the median value for that parameter. To form the aggressive case, the best value from the published results is chosen for each parameter. The average case is similar to the ITRS 2007 projection for PCM technologies [69] and the aggressive case mimics PCM device improvement as technology advances.

As the framework is not a full-system simulator, a memory management model is developed based on source code of several key memory management routines in the Linux Kernel 2.16.15.5 and integrated it into the simulator to mimic the OS page allocation/deallocation/migration and page table lookup. Meta-data structures are also implemented, such as page descriptors and \texttt{vm\_area} structure in our framework. As the simulator doesn’t support system calls, our memory management is triggered by tracking the memory reference footprints to previously unreferenced pages, similar to page fault expectation handling by OS.

Moreover, detailed thermal analysis is performed to understand the impact of PCM technology on thermal constrained 3D architecture. The temperature-modeling tool used is Hotspot4.0 [70]. The floorplan of the core layer is shown in Figure 3-11 B). The power trace input to Hotspot is generated by Wattch and DRAMsim power model. The layer configuration
follows the physical dimension shown in Figure 3-11 A). The bulk silicon is modeled as one layer of 800μm in thickness. The thermal parameters of materials used are the same as those in [1] and Hotspot’s default heat spreader and heat sink models are set for thermal simulations.

Results

This section presents the experimental results, including: 1) power savings achieved by PCM technology; 2) performance impact of PCM-based memory design; 3) life span extension by using BCH with varied error correction capability and OS wear-leveling; and 4) thermal stress and performance improvement due to the relieved thermal constraints in 3D design.

Power Saving of PCM Technology

“3D Planar 2G-DRAM (NC)”, which is a planar on-chip 2G-Byte DRAM with normal cooling (NC) solution, is used as our baseline. It is observed that with normal cooling capacity, the peak temperature of all workloads exceeds 85°C, which requires a 32ms refresh interval [59] to ensure reliable operation for DRAM. The “3D Planar 2G-DRAM (AC)” shares the same design configuration as the baseline but with aggressive cooling (AC) capacity so that the on-chip temperature is below 85°C. A 64ms DRAM refresh interval is necessary in this case [59].

The “True 3D 2G-DRAM (NC)” is similar to the “3D Planar 2G-DRAM (NC)”, except that the DRAM is designed in a true-3D fashion. For all PCM-based designs, the approach of programming current adaptation is used. The “3D Planar 2G-PCM (AC)” and “True 3D 2G-PCM (NC)” are planar and true-3D PCM with aggressive and normal cooling capacity respectively. The “3D Planar 2G-PCM (NC)” and “True 3D 2G-PCM (AC)” configurations are not presented because the overall impact of temperature and TSVs on PCM power is captured by “True 3D 2G-PCM (NC)” design. The “Hybrid” is the proposed scheme, which uses normal cooling capacity with low cost.
A breakdown of memory power (normalized to the baseline case) is first presented on workload H4 to illustrate the impact of different on-chip memory designs. The overall background power includes PRE_PDN, PRE_STBY, ACT_PDN and ACT_STBY and REF [71]. To highlight the impact of elevated temperature (due to 3D die stacking) on DRAM refresh power, we separate the refresh power (Refresh) from the rest of the background power (Bg). The contribution of Bg (exclude Ref), Refresh, Activate, RD and WT to the overall memory power is 21%, 6%, 40%, 16%, 17% respectively in the baseline case. Figure 3-12 shows that the refresh power reduces dramatically (e.g. by a factor of 3X) when the long refresh interval is used, while the PCM-based memory does not require refresh due to its non-volatile nature. This non-volatile nature also leads to a lower background power, shown by the first set of bars. Moreover, in contrast to DRAM, the activation of PCM does not need to charge, discharge and sense all bit lines that connect to a row of cells, which results in a significant active power saving (i.e. more than 50%), as shown by the third set of bars. The last set of bars in Figure 3-12 shows that the true 3D PCM implementation reduces memory write power (WT) by 22% compared with a 3D planar PCM design, which is largely due to the increased number of TSVs used in the design. Although 3D planar PCM consumes 40% more write power than DRAM, its true-3D design significantly reduces the write power overhead to 10%.

Figure 3-13 shows the percentage of pages that have ever been migrated to DRAM partition for each workload in the hybrid system. Note that page classification is based on modification frequency. Thus, the percentage of pages migrated primarily depend on read/write pattern rather than memory footprint size. As one can see, on average only 11% of pages have ever been migrated to DRAM and we observe 83% of memory accesses are performed on PCM. This observation shows SPEC multiprogramming workloads sufficiently stress PCM. Figure 3-
14 shows the overall memory power of different designs across all experimented workloads. For DRAM-based design, the reduction of temperature-induced refresh power from “3D Planar 2G-DRAM (NC)” to “3D Planar 2G-DRAM (AC)” contributes to an average 8% decrease in the total DRAM power. Although the true-3D implemented DRAM achieves an average 15% power reduction due to the reduction in wire-length, the charging and discharging of a large number of bit lines cannot be avoided. The “True 3D 2G-PCM (NC)” reduces the memory power by 55% compared to the baseline case and achieves the best power efficiency on the majority of all experimented workloads. The significantly reduced background and activation power and the eliminated refresh power are the primary contributors. Compared to the “True 3D 2G-PCM (NC)”, our proposed scheme (“Hybrid”) also achieves 54% power savings, but introduces a small power overhead for median- and low- miss workloads due to the added small DRAM partition. For high-miss workloads, such as HM1-HM4, “Hybrid” achieves slightly more power savings than “True 3D 2G-PCM (NC)”. This is because our OS page allocation scheme migrates hot-modified pages to DRAM where write operations consume less power than they do on PCM. Figure 3-15 presents performance comparison results. Due to the relatively slower read speed of PCM as compared to “True 3D 2G-DRAM (NC)”, the “True 3D 2G-PCM (NC)” results in 9% performance degradation on average. By using the proposed OS page allocation and migration, our hybrid design reduces the performance degradation to 6% on average.

The proposed hybrid design achieves 51% energy reduction, slightly worse than the 54% savings in power, due to the performance overhead. Furthermore, the impact of PCM technology trend on power, performance and energy is shown in the Table 3-4. The ITRS 2007 is the proposed design that uses PCM parameters from International Technology Roadmap for Semiconductors 2007 Edition [69]. As PCM technology improves, the performance of PCM will
be comparable to DRAM, while the power consumption is expected to be further reduced. This reduction is primarily contributed by decreasing the required programming current achieved through device and material improvements.

The proposed design is further compared to several other DRAM-based power management techniques, namely, Smart Refresh [6], Queue-Aware Power Down [72], Min-Rank [73] and Flash-based disk caches [18], which achieve 8.1%, 20% 34% and 41% power saving respectively according to our simulations. The proposed scheme achieves at least 13% more power savings. Since the proposed PCM/DRAM hybrid memory architecture contains a DRAM partition, these proposed techniques can be employed to the design in this research to achieve further power saving. Furthermore, [74] and [75] propose two architectural techniques to reduce PCM power. [74] proposes to use narrow rows and multiple buffers to improve write coalescing and perform partial writes. [75] takes advantage of redundant bit-writes to eliminate unnecessary writes to PCM and perform dynamic memory mapping at memory controller to achieve wear-out leveling. These two techniques can be applied to our design to achieve additional 5% power saving.

**Endurance Enhancement**

To estimate the wear-out induced failures, we run each workload repeatedly and track the number of writes to each bit. $1\times10^5$ instead of $1\times10^8$ writes is used in our lifetime estimation due to the extremely long simulation time to collect $1\times10^8$ writes to any PCM cell. With this accelerated estimation method, when a bit has more than $1\times10^5$ writes on it, it is marked as a failed cell. A memory access failure is defined as the number of failed bits accessed in a memory reference is larger than the number of errors that can be corrected. The lifetime of PCM-based memory is estimated as the number of cycles elapsed before the first memory access failure occurs.
Figure 3-16 shows the lifetime improvement of the proposed technique (hybrid+OS paging+BCH). All results are normalized to the lifetime of the PCM-only memory system using a random page allocation (PCM+OS random page allocation). The results of the hybrid PCM/DRAM architecture that relies on OS paging scheme only (hybrid+OS paging, i.e. no BCH) and BCH only (hybrid+BCH, i.e. no page migration to DRAM partition after initial allocation on PCM) for reliability enhancement are also presented. The “hybrid+OS paging” scheme achieves 4X endurance enhancement (the geometric mean across all simulated workloads). Note that the lifetime is extended by 16X for high miss workloads. This is because the applications in the high-miss category stress memory more intensively than others, leading to a shorter lifetime on the baseline design. It is observed that several applications (e.g. mcf, art and apsi) frequently modify a small region of memory. By reallocating those pages from PCM to DRAM partition, the proposed technique can substantially reduce the number of writes to PCM memory and extend the lifetime of the entire memory system. The “hybrid+BCH” scheme allows 28X endurance improvement. The proposed techniques, which combine BCH and OS paging, extend the life span of the memory system by 114X. The BCH performance penalty increases with the number of errors corrected. As shown in Figure 4-17, compared with “hybrid+BCH”, our proposed techniques (hybrid+BCH+OS wearout aware page alloc) have the ability of achieving wear-leveling, which reduces the number of errors that needs to be corrected per memory access, resulting in 10% performance improvement in geometric mean of all experimented workloads.

**Thermal Relief and Performance Benefit**

Figure 3-18 illustrates the temperature distribution (on the workloads H3, i.e. fma3d+lucas+apsi+wupwise) of the hottest layer (processor layer) of a 3D DRAM-only system in contrast to that of our proposed design. Figure 3-20 shows the peak temperature reduction
achieved by replacing the conventional DRAM based memory with the proposed PCM-based design across all experimented workloads. The peak temperature is the sampled maximum temperature across all layers and throughout the entire execution. The temperature reduction ranges form 2.7°C to 4.25°C with an average of 3.3°C (as shown in Figure 3-20). Arising from reliability concerns, the temperature constraint on a 3D-chip limits the maximum operating frequency of the system. In Figure 3-19, an example of the peak temperature as a function of frequency is presented. Given a thermal constraint, such as 100°C, it places a lower limit on the operating frequency in the proposed hybrid PCM/DRAM architecture. Figure 4-19 shows that the maximum frequency allowed increases by 300 MHz, which translates to an execution speedup of 1.07 on H3 over the DRAM-only design. Figure 3-20 shows the execution speedup across all workloads. As can be seen, an average speedup of 1.07 is achieved. The results indicate that the lower power of PCM technology is capable of alleviating the thermal constraint of 3D technology and achieving additional performance gains.

**Memory-Intensive Workload Results**

Figure 3-21 shows the memory power consumption for each design. The proposed hybrid design can also effectively reduce memory power by 50% across all memory-intensive workloads with very large memory footprints. Whereas, the average memory access latency for the hybrid design increases by 18% compared to DRAM design, shown in Figure 3-22. As the performance of memory-intensive application is primarily determined by memory access latency, average memory access latency is used to estimate the performance impact of PCM due to the limitation of our simulation infrastructure. Although both PCM and DRAM have approximately the same cell size, adopting the Multi-Level Cell technology substantially increases the density of PCM, allowing a larger memory capacity with the same die area.

“Hybrid w/ MLC for PCM” in Figure 3-22 shows the average memory access latency of a hybrid
design with MLC technology applied on PCM partition, resulting in a hybrid memory with 4G effective PCM and 128MB DRAM. As can been seen, “Hybrid w/ MLC for PCM” is able to achieve a 28% and 15% reduction in average memory access latency over “Hybrid” and “True 3D 2G-DRAM (NC)” respectively.

**Related Work**

In the past, there have been many studies on DRAM power management at both the architecture and OS levels. At the architecture level, the variation of retention-time among cells is exploited in [14]. The OS-based DRAM power management approaches take advantage of low power modes implemented in today’s DRAM chips and maximize its power benefit with optimization techniques in [15]. A common theme of the above approaches is to reduce standby or background power by sending chips to low-power states. However, these techniques are unable to entirely eliminate it because normally they cannot power down all DRAM banks. This work is unique in that it explores a new type of memory technology, PCM, which doesn’t require data retention power. In addition, the characterizations presented in this research show that PCM is more suitable for 3D-stacked memory application compared to others. Similar to prior OS-based approaches, a paging-based scheme is used in this work for the purpose of extending endurance of PCM. eDRAM has been used for low power on-chip memory. However, eDRAM (like conventional DRAM) requires capacitor to store charges for retaining information. The retention time decreases with technology scaling down. Furthermore, the elevated temperature in 3D exacerbates the leakage issue.

Due to their low power features, nonvolatile memories such as Flash, MRAM and FeRAM are being increasingly used to build storage systems and on-chip structures. [76][77] evaluates performance and power of 3D stacking cache using MRAM. [78][79] further evaluates a hybrid cache design, which takes advantage of the best characteristics of each nonvolatile memory
technology. This work focuses on exploring PCM in main memory system design and propose OS support to efficiently address the disadvantages of using PCM in such application. For main memory design using alternative technologies, [63] first proposed integrating a Flash-based disk cache into memory hierarchy of server platforms to save memory power. The PCM technology introduced by this work is superior to Flash in terms of performance and lifespan. [74][75] are the first two architecture level studies on using PCM to implement main memory. This study differs from these works in two ways: 1) this study evaluates PCM in the context of true 3D-stacked memory architectures, while their designs are limited only to the planar chips; and 2) this work proposes both architecture and OS-level endurance improvement techniques, whereas their schemes are built only on the architecture layer. Techniques proposed in [74][75] are orthogonal to our design. By incorporating their schemes into our design, an additional 5% power saving is achieved.
Table 3-1. Baseline machine configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width</td>
<td>4-wide fetch/issue/commit</td>
</tr>
<tr>
<td>IQ, ROB, LSQ</td>
<td>64 Issue Queue, 96 ROB entries, 48 LSQ entries</td>
</tr>
<tr>
<td>TLB</td>
<td>128 entries(ITLB), 256 entries(DTLB), 4-way, 200 cycle</td>
</tr>
<tr>
<td>Branch Pred.</td>
<td>2K entries Gshare, 10-bit global history, 32 entries RAS</td>
</tr>
<tr>
<td>I/D L1 Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle</td>
</tr>
<tr>
<td>Integer ALU</td>
<td>4 I-ALU, 2 I-MUL/DIV, 2 Load/Store</td>
</tr>
<tr>
<td>FP ALU</td>
<td>2 FP-ALU, 2 FP-MUL/DIV/SQRT</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>shared 2MB, 8-way, 128 Byte/line, 12 cycle</td>
</tr>
<tr>
<td>Memory</td>
<td>DDR2, 2GB, 8 banks, 667MHz, open page</td>
</tr>
</tbody>
</table>

Table 3-2. Workloads

<table>
<thead>
<tr>
<th>Type</th>
<th>Workloads</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>H1(mcf, art, apsi, wupwise)</td>
</tr>
<tr>
<td></td>
<td>H2(art, art, fma3d, apsi)</td>
</tr>
<tr>
<td></td>
<td>H3(fma3d, lucas, apsi, wupwise)</td>
</tr>
<tr>
<td></td>
<td>H4(fma3d, apsi, wupwise, equake)</td>
</tr>
<tr>
<td></td>
<td>M1(swim, swim, gzip, gzip)</td>
</tr>
<tr>
<td></td>
<td>M2(swim, gzip, applu, bzip)</td>
</tr>
<tr>
<td></td>
<td>M3(gzip, gzip, gcc, ammp)</td>
</tr>
<tr>
<td></td>
<td>M4(gzip, applu, gap, ammp)</td>
</tr>
<tr>
<td>Moderate</td>
<td>L1(parser, facerec, vortex, galgel)</td>
</tr>
<tr>
<td></td>
<td>L2(mgrid, mgrid, facerec, facerec)</td>
</tr>
<tr>
<td></td>
<td>L3(mgrid, facerec, twolf, mesa)</td>
</tr>
<tr>
<td></td>
<td>L4(vpr, vpr, facerec, twolf)</td>
</tr>
<tr>
<td>Low</td>
<td>HM1(mcf, art, swim, gzip)</td>
</tr>
<tr>
<td></td>
<td>HM2(mcf, fma3d, swim, applu)</td>
</tr>
<tr>
<td></td>
<td>HM3(fma3d, lucas, applu, bzip)</td>
</tr>
<tr>
<td></td>
<td>HM4(art, lucas, gzip, bzip)</td>
</tr>
<tr>
<td></td>
<td>ML1(swim, gzip, mgrid, vpr)</td>
</tr>
<tr>
<td>High -Moderate</td>
<td>ML2(swim, applu, mgrid, parser)</td>
</tr>
<tr>
<td></td>
<td>ML3(applu, bzip, parser, facerec)</td>
</tr>
<tr>
<td></td>
<td>ML4(gzip, bzip, vpr, facerec)</td>
</tr>
<tr>
<td></td>
<td>4 X bt</td>
</tr>
<tr>
<td></td>
<td>4 X cg</td>
</tr>
<tr>
<td></td>
<td>4 X ft</td>
</tr>
<tr>
<td>NAS Parallel Bench</td>
<td>4 X lu</td>
</tr>
<tr>
<td></td>
<td>4 X mg</td>
</tr>
<tr>
<td></td>
<td>4 X sp</td>
</tr>
<tr>
<td>Triad</td>
<td>4 X Triad</td>
</tr>
<tr>
<td>Qsort</td>
<td>4 X Qsort</td>
</tr>
<tr>
<td>Kmean</td>
<td>4 X Kmean</td>
</tr>
</tbody>
</table>
Table 3-3. DRAM/PCM timing and power parameters

<table>
<thead>
<tr>
<th></th>
<th>DRAM</th>
<th>PCM</th>
<th>Aggressive</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average</td>
<td>60</td>
<td>9.9[80]</td>
</tr>
<tr>
<td>tRCD</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tRAS</td>
<td>40(2D)/ 27(3D) (1)</td>
<td>NA*</td>
<td>NA*</td>
</tr>
<tr>
<td>tCAS</td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tWR</td>
<td>15</td>
<td>150/250</td>
<td>50/120[69]</td>
</tr>
<tr>
<td>tRP</td>
<td>15</td>
<td>NA*</td>
<td>NA*</td>
</tr>
<tr>
<td>tCMD</td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Active recharge</td>
<td>135</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Precharge power down</td>
<td>8</td>
<td>77(2)</td>
<td></td>
</tr>
<tr>
<td>Precharge standby</td>
<td>70</td>
<td>62(4)</td>
<td></td>
</tr>
<tr>
<td>Active power down</td>
<td>40</td>
<td>0(3)</td>
<td></td>
</tr>
<tr>
<td>Active standby</td>
<td>75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read</td>
<td>275</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write</td>
<td>255</td>
<td>325(5)†</td>
<td>263(6)†</td>
</tr>
<tr>
<td>Refresh</td>
<td>280</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1)27ns = 40×(1-32%), 32% improvement in tRAS for 3D implementation [1]
(2)77 = standby current (62) + Decoder current (15) obtained by using CACTI 6.0
(3)0 power consumption when master on-off switch for the PCM is turned off.
(4)62 accounts for the current due to clock signal and peripheral circuit leakage. It is calculated according to a current diagram from Micro technology nodes [71].
(5)325 = DRAM Write Current (255) + PCM Write Driver (~70mA[81])
(6)263 = DRAM Write Current (255) + [PCM write current (100uA/bit[80])*16 bit[81]] / PCM Write Driver Efficiency (20% [81])
† We conservatively use the value of RESET current for both RESET and SET to estimate power consumption, although SET use a smaller current
* doesn’t apply to PCM

Table 3-4. The impact of PCM technology trend on power and performance (Normalized to baseline)

<table>
<thead>
<tr>
<th></th>
<th>Power</th>
<th>Performance</th>
<th>Energy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average</td>
<td>46%</td>
<td>93%</td>
<td>49%</td>
</tr>
<tr>
<td>ITRS 2007</td>
<td>46%</td>
<td>94%</td>
<td>49%</td>
</tr>
<tr>
<td>Aggressive</td>
<td>34%</td>
<td>98%</td>
<td>35%</td>
</tr>
</tbody>
</table>
Figure 3-1. The basic structure of a PCM cell*
*The SET and RESET states of the PCM correspond to a stored binary “1” or binary “0”.

Figure 3-2. The programming pulses of PCM[21]

Figure 3-3. Sub-array architecture of PCM memory
Figure 3-4. The overall architecture of a PCM device and the illustration of a successful RESET and the one dimensional temperature profile [57]

Figure 3-5. The temperature dependence of a PCM device’s programming power

Figure 3-6. 2D planar PCM prototype

Figure 3-7. PCM constructed in true-3D organization
Figure 3-8. A) An overview of the proposed hybrid PCM/DRAM memory system architecture and B) The 3D integration of processor and memory system

Figure 3-9. Latency overhead with the increased number of corrected errors

Figure 3-10. Wear-out aware OS page allocation.
Figure 3-11. A) The Cross-section view of the simulated 3D (Left) and B) the floor-plan of processor layer (unit mm) (Right)

Figure 3-12. A breakdown of memory power (workload: H4).

Figure 3-13. The percentage of pages migrated to DRAM
Figure 3-14. A comparison of overall memory power

Figure 3-15. The performance impact of PCM memory design

Figure 3-16. A comparison of endurance improvement

Figure 3-17. The impact of wear-out aware page allocation on BCH performance

Figure 3-18. An illustration of on-chip temperature (workload: H3)
Figure 3-19. An example of thermal constraint on maximal allowed frequency (workload:H3)

Figure 3-20. The peak temperature reduction due to the PCM’s power reduction and the execution speedup enabled by the reduced peak temperature

Figure 3-21. PCM power impact for memory-intensive workloads

Figure 3-22. PCM performance impact for memory-intensive workloads
CHAPTER 4
CHARACTERIZING AND MITIGATING THE IMPACT OF PROCESS VARIATIONS ON PHASE CHANGE BASED MEMORY SYSTEMS

Among the desirable characteristics of PCM (e.g. non-volatility and very low standby power), superior scalability is the most attractive. As memory density increases with each smaller generation, the volume of PCM’s phase change material (e.g. Ge₂Sb₂Te₅) shrinks as well, providing a truly scalable solution. This will allow PCM to scale down for the next several generations of processing technology and have no physical limits until at least the 20nm technology node [20]. Although advanced process technologies provide smaller and faster devices (through aggressive feature size scaling), difficulties in manufacturing control have resulted in significant variability in the fabricated devices. At the chip-level, process variation can lead to performance discrepancies between dies. Process variation also exists at fine-grain levels and manifests as correlations among within-die devices. Transistor variability significantly reduces chip operating frequency and increases power consumption [82][83][84]. Recently, there have been increased interest in characterizing the impact of Process Variation (PV) on processor performance/power and developing circuit- and microarchitecture- level techniques to mitigate the deleterious effect of PV [85][86][87][88]. These studies, however, all focus on conventional, CMOS-transistor based storage cells and combinational logic. As PCM is becoming a promising technology for nano-scale non-volatile memory system design, architects need to understand the system-level implication of PV on PCM memory and develop cost-effective PV mitigation techniques. Towards this goal, this research provide the first study on the types and sources of process variations in PCM design and analyze their impact on PCM endurance and programming power. Circuit-, microarchitecture- and system- level optimizations are proposed to overcome the deleterious impact of PV on PCM. The synergy of the proposed cross-layer approaches...
yields cost-effective PV-aware design methodologies for nano-scale non-volatile memory using emerging phase change technologies.

**Process Variations in PCM: The Source and Impact**

In this section, several PCM design parameters that are subject to process variation are described and analytical models to quantify how process variation affects $I_{\text{RESET}_{\text{min}}}$ are presented. The impact of PV on PCM power and endurance is characterized due to the variability of $I_{\text{RESET}_{\text{min}}}$. Due to space limitations, only the analysis for the RESET operation is presented in this section, but a similar characterization procedure has been applied to the SET operation. The analysis shows that $I_{\text{SET}_{\text{min}}}$ is about 70% of $I_{\text{RESET}_{\text{min}}}$ (consistent with the conclusion draw from experiments in [81]). Note that the variation in $I_{\text{RESET}_{\text{min}}}$ will affect $I_{\text{SET}_{\text{min}}}$ proportionally as SET is the reverse operation of RESET.

**PCM Design Parameters subject to Process Variation**

(1) Variations in Physical Dimensions of PCM Devices

Process variation affects various PCM physical dimension parameters (e.g. the thickness of the phase change material layer, the height of the heater, the contact size etc.), resulting in variability in PCM electrical characteristics such as $I_{\text{RESET}_{\text{min}}}$. Among these, the impact of the Bottom Electrode Contact Diameter (BECD) size on $I_{\text{RESET}_{\text{min}}}$ has been experimentally demonstrated in [22]. The decreased BECD increases local current density and joule heating in GST, resulting in a melting down of the adjacent crystalline material more efficiently. This allows the use of a smaller $I_{\text{RESET}_{\text{min}}}$ (e.g. below the nominal value) to program a memory cell without causing a failure. On the contrary, a larger BECD reduces current density and joule heating, requiring a higher magnitude of current to reach the desired temperature. As the device feature size continues to scale down, it becomes increasingly difficult to fabricate PCM cells.
with uniform contact size. This makes achieving a uniform programming current across all PCM cells increasingly challenging.

Another design parameter that affects PCM write current is the thickness of phase change material (ThickGST). Unlike BECD, the variability of ThickGST affects $I_{\text{RESET}_{\text{min}}}$ indirectly. When the ThickGST exhibits a positive variation (i.e. a thicker phase change material layer), the SET resistance is higher than the average because it is proportional to the thickness of GST. The RESET resistance, on the other hand, is primarily determined by the volume of crystalline material melted at RESET. Note that the PCM read operation is performed by sensing the resistivity of the cell. The reduced SET and RESET resistance margin, if smaller than a minimum threshold, is likely to lead to a false read. To avoid this, the difference between SET and RESET resistance needs to be enlarged for correct sensing. To achieve this goal, a higher $I_{\text{RESET}_{\text{min}}}$ (i.e. above nominal value) is required to transform a larger volume of GST material from crystalline state to amorphous state. On the other hand, a lower $I_{\text{RESET}_{\text{min}}}$ is sufficient when ThickGST is reduced.

Variations in the thickness of heater (ThickHEATER), which result in variability in its resistance, also have a strong influence on $I_{\text{RESET}_{\text{min}}}$. As a high resistance is desirable for heat generation ($P=I^2R$), a thinner layer of heater is likely to demand a higher $I_{\text{RESET}_{\text{min}}}$ to reach the desired temperature for PCM phase transition.

(2) Variation in Transistor Parameters within PCM cells

Transistors in PCM peripheral circuits control the amount of programming energy that can be delivered to memory cells. Both the magnitude and width of the write current is controlled by transistors (shown in Figure 4-5). Since PV in the transistor gate length can change the effective driving capability of these transistors, PV affects PCM write energy by changing the magnitude
of the write current. Precise control of the write energy is crucial for successful programming operations. This is because the actual resistance of a programmed PCM device is sensitive to the write energy. Moreover, the variation in $V_{th}$, which affects the speed of transistors, will affect PCM programming energy by changing the width of write current pulse. However, $V_{th}$ does not affect the amplitude of the programming current.

**Characterizing the Effect of Process Variations on PCM Programming Current**

A 2GB PCM chip consisting of 8 banks (the layout is the same as the one shown in Figure 4-2) is modeled in this research and SPICE simulations and analytical models are used to quantitatively evaluate the effect of process variation on PCM programming current $I_{\text{RESET \_min}}$. It is assumed that the $I_{\text{RESET \_min}}$ is 0.3mA when there is no PV. To model the impact of transistor parameter variation on the magnitude and width of PCM programming current pulses, SPICE models are built for PCM write electrical path. The physical MOSFET model used is BSIM [43]. In addition, a one-dimensional heat conduction model [57] is used to quantify the effect of variation in the physical dimensions of phase change material on PCM programming operations.

Figure 4-5 illustrates the electrical path of PCM cell write operations. Transistors that have a direct impact on the magnitude and width of the programming current are marked. Among these transistors, transistor 1 (located in PCM cell array) determines the magnitude and transistors 2 and 3 affect the width of the programming current. As transistors 2 and 3 are physically located close to each other (both in the PCM peripheral circuit), they are likely to share a similar variation due to correlation. Therefore, we assume variations exhibit correlation between transistors 2 and 3, while variations between transistor 1 and transistors 2 and 3 are independent. The results presented in Figures 4-6 and 4-7 show the PV effect when a transistor parameter (e.g. gate length) of one group varies and the other remains constant. As shown in Figure 4-6, the widths of the write current pulses vary with the transistor gate length. The
variation is around 1ns, which is only 2% of a typical current width. It is also observed that the width of write current is affected by the variation of $V_{th}$, but the resulting variation in current width is very small, within 5%. However, the magnitude of the current shows a considerable variation as the transistor gate length varies. As shown in Figure 4-7, the magnitude of current pulses delivered to the PCM cells decreases as the gate length increases because of the increased transistor on-resistance. Consequently, transistors with longer gate length may not be able to deliver sufficient current to successfully program a cell, while transistors with shorter gate length are likely to over program the PCM cell.

To model the impact of PCM physical dimension parameter variability on $I_{\text{RESET}_{\min}}$, a heat conduction model[57] is used, which links temperature to the minimal programming currents required for performing successful RESET operations. This heat conduction model captures the flow of heat in the PCM device that is insulated everywhere except at the two ends, which connect to bit-line and access transistor. The model takes both device physical dimensions and properties of PCM material as inputs and produces an estimated $I_{\text{RESET}_{\min}}$. This allows us to quantify the impact of variation in the physical dimensions of PCM devices on $I_{\text{RESET}_{\min}}$.

The variation in $I_{\text{RESET}_{\min}}$ is characterized by varying each physical dimension parameter individually. The results are shown in Figures 4-8, 4-9 and 4-10. These figures also plot the RESET current distribution across memory cells in the modeled PCM chip. As can be seen, among all of the parameters, PCM write current is the most sensitive to variation in BECD and ThickHEATER. As shown in Figure 4-8, the deviation of $I_{\text{RESET}_{\min}}$ can be as much as 32% due to the variation of BECD. Smaller contact area allows lower current and this observation is consistent with [89], which designs low power PCM using a ring type contact to achieve a minimum programming current. Moreover, the variation in ThickHEATER can affect $I_{\text{RESET}_{\min}}$.
by as much as 46%, shown in Figure 4-9. A thinner heater layer results in a higher $I_{\text{RESET\_min}}$. This is because thermal energy generated by a low resistance Joule heater (i.e. $q_{\text{Joule}} = I^2 R t$) is limited. In addition, a thinner heater layer is unable to provide a high temperature environment under the GST layer. This results in a large temperature gradient between the GST layer and the material below it. Instead of concentrating on heating the GST material, the generated thermal energy dissipates. As shown in Figure 4-10, $I_{\text{RESET\_min}}$ is not sensitive to the thickness of the GST layer (primarily consist of crystalline material) and the variation of $I_{\text{RESET\_min}}$ is only about 1%. The volume of amorphous material melted by the RESET current is the primary contributor to the high RESET resistance, while the crystalline state is a very small contributor. As the thickness of the GST layer varies, the RESET resistance is affected slightly. Therefore, the resistance margin still satisfies the requirement and $I_{\text{RESET\_min}}$ doesn’t exhibit a substantial variation.

To quantify the aggregated impact of parameter variation on the PCM programming current, each variation source is modeled using the methods described above and its statistical distribution is generated for each PCM chip, as is shown in Figure 4-11(A). Then the generated PCM physical dimension parameter statistics are used as inputs to the one dimensional heat conduction model to compute the $I_{\text{RESET\_min}}$ for each memory cell (Figure 4-11(B)). The variability of $I_{\text{RESET\_min}}$ across the PCM chip is shown in Figure 4-11(C) and its histogram is shown in Figure 4-11(D). Note that Figure 4-11(C) and 4-11(D) only show the current profile for one of 400 PCM chips that were simulated. The PCM RESET current distribution shows a 40% variation when the variation of all parameters are taken into account. This variation of $I_{\text{RESET\_min}}$ is caused by the overall parameter variability and is smaller than the one due to varying the
ThickHEATER alone. This is because the variation effect contributed by other sources is small and the overall impact of parameter variability on programming current is reduced.

**PV Induced PCM Programming Power Overhead and Reliability Degradation**

Due to the variation in the PCM design parameters, the required minimal programming current varies across the memory cells. In a conventional design, to ensure the successful programming of a PCM cell, the current generated for write operations has to be tailored to the worst case. Consequently, the write driver functional block has to raise its voltage to a maximum level in order to deliver sufficient current for proper operation of all of the PCM cells. This not only consumes 96% more write power (shown in Figure 4-18) than what is necessary, but also exposes 36% of the PCM cells (e.g. the areas shown in red and yellow in Figure 4-11(C)) to a much higher current than needed. Such over programming can physically degrade the endurance of phase change material, leading to reduced reliability. The endurance test performed in[90] shows that the lifetime of a PCM device is a function of the programming current pulse energy. The experimental data (Figure 4-12, obtained from [90]) shows a power law relationship between cycle lifetime and programming energy. The dependency of endurance on the magnitude of the RESET current indicates that overheating the cell with higher than necessary current leads to a reduced life cycle. As shown in Figure 4-11, to tolerate PV, the RESET current increases from 0.3mA (the mean value) to 0.42mA (the maximum value) due to conventional design. Assuming the energy value of 1 corresponds to the 0.3mA RESET current, the energy of RESET increases by 96% due to conventional design and this energy increase results in a 50X reduction in endurance. Note that we assume an endurance of 1E08, as projected by ITRS 2007[69], for PCM cells that have a nominal RESET current. The write endurance requirement for a computer system can be estimated using the following equation[16],

```plaintext
e_{write} = \frac{E_{RESET}}{1E08} 
```
where $E$ is the cycling endurance, $T_{life}$ is life expectancy of the system, $B$ is memory bandwidth, $\alpha$ is wear-leveling efficiency and $C$ is the system memory capacity. Assuming a typical server with 1GB/sec bandwidth, 0.1 wear-leveling efficiency and 4GB PCM, the estimated lifetime of PCM memory is about 8 years, when there is no PV. This estimated lifetime can be significantly reduced, to as little as 2 months ($8\text{ years} / 50 \approx 2\text{ months}$), due to PV-induced over programming. PV-aware PCM design is highly desirable as a means to overcome the abovementioned power and reliability disadvantages. Note that although the power and endurance of PCM is affected by PV, the write latency of a PCM cell is largely independent of PV. This is because the delay in transition between crystalline and amorphous states largely depends on the property of phase change material [91].

**Mitigating Process Variation Impact on PCM Power and Reliability**

In this Section, cross-layer techniques that span circuit-, microarchitecture- and system-levels are proposed to cost-effectively mitigate the deleterious impact of PV on PCM power and reliability.

**Variation-aware Programming Current Provision**

The proposed variation-aware PCM programming current provision technique addresses the variability issue by adaptively tuning the voltage level, which makes the programming current adjustable. Instead of using a uniform current for programming all PCM cells, the proposed scheme provides multiple current magnitudes that can be chosen. The entire PCM array is divided into multiple domains and each domain can select the lowest current magnitude that is sufficient to successfully program all cells within that domain. To achieve this goal, both post-fabrication tuning and run-time adaptation mechanisms are used.
A typical PCM write driver circuit, shown in Figure 4-13 (A), adopts Dickson Charge pump design [92] to generate a voltage level that is high enough to ensure successful programming on any cell across the entire memory. When the write enable single is activated, the charge pump functional block starts pumping and continues until the target voltage level is reached. After that, the memory cell transistor is turned on and a write current flows through the cell, leading to the phase change in the cell. Breaking the charge pump down into multiple steps is proposed shown in Figure 4-13(B). The voltage produced after the initial pump node, \( WD_{\text{Pump}} \), is high enough to provide current for programming some of the cells. Other cells may demand higher voltage/current due to process variation. Functional blocks such as \( Extra_1 \) and \( Extra_2 \) can provide additional charge to further increase the voltage level at

\[
\Delta V = \frac{C}{C+C_s} V_\phi - \frac{I_{\text{out}}}{(C+C_s) f_{\text{osc}}}
\]

per step (\( C_s \): stray capacitance, \( V_\phi \): the magnitude of anti-phase voltage, \( f_{\text{osc}} \): operating frequency of charge pumps, \( I_{\text{out}} \): driving capability). In this study, 8 voltage levels is used with \( \Delta V = 0.4V \), which provides a current difference of 0.03mA between the two tuning levels. It is assumed that \( V_\phi = 1.1V \), \( f_{\text{osc}} = 1GHz \), and \( I_{\text{out}} = 20mA \). The overall delay introduced by the additional 8 charge pump stages is less than 8ns (\( 1 / 1GHz \times 8 \text{ stages} = 8 \text{ ns} \)).

To employ the proposed current provision technique, post-fabrication calibration is used to identify the minimum voltage boost required for successful write operation. This post-fabrication tuning information is then stored in a small flash memory as part of the peripheral circuit and is used to guide the run-time adaptation of voltage boosting. Given a write address, the write driver block performs a lookup in the storage containing post-fabrication (area and power overhead discussed later in this section) tuning information to find an appropriate voltage level. Then a number of charge pumps are enabled according to the level of output voltage.
An important design decision of the proposed technique is the tradeoff between tuning resolution (i.e. the number of domains) and the introduced overhead. If the tuning is very fine-grained (e.g. per bit), the circuit overhead (e.g. the memory space due to store tuning information) may offset the achieved benefit. On the other hand, if the tuning is very coarse-grained (e.g. per die); it will be unable to mitigate process variation induced power and reliability overheads. Process variability is a type of layout-dependant variation through which nearby devices exhibit spatial correlations. Current memory design usually adopts a hierarchical organization consisting of sub-arrays, sub-banks and banks. As a result, memory cells residing in the same sub-array are likely to exhibit similar variability and \( \frac{I_{\text{RESET \_ min}}}{I_{\text{SET \_ min}}} \). In this study, a tuning resolution of one memory sub-array is chosen, which is 4-MB in size. This design choice requires a 1KB storage for post-fabrication tuning information with a die area overhead of 0.01\( \text{mm}^2 \), an additional power consumption of 0.01mW and an access latency of 2ns at 45nm technology. Another critical design choice is the voltage difference between two adjacent steps. Fine-grained voltage tuning with a large number of steps allows for a more accurate current provision. However, it will also incur significant die area overhead. This is because the charge pump is broken into many stages and this demands more circuits to implement the entire charge pump block. After performing sensitivity analysis and estimating the die area overhead, it is found that a number of 8 voltage levels is sufficient to achieve the goal with a die area overhead of 0.23\( \text{mm}^2 \) (less than 0.1% of total PCM chip area).

**Adaptive Data Comparison Write Using Page Classification**

The scheme proposed in the previous section allows the tuning of programming current for individual domains. However, it is incapable of reducing PV-induced power and endurance overhead on cells whose programming currents are larger than the nominal value within each domain. In this section, novel techniques to mitigate PV-induced overhead are proposed by
taking advantage of PCM’s non-destructive read feature. The goal is to reduce the number of writes to cells that are negatively affected by PV within each domain. Unlike DRAM, a read to a PCM cell does not destroy the stored data. Therefore, no write back to PCM cell is needed after a read operation. PCM writes are required to update the memory array only if a dirty cache line is evicted. A partial writes scheme is proposed in [74] to eliminate unnecessary word writes via tracking dirty data in the cache system and only writing back dirty data in the evicted cache lines to the PCM-based memory. Note that it is assumed that the size of each word is equal to 4B.

In this work, an alternative technique (i.e. adaptive data comparison write) is proposed, which can effectively eliminate redundant bit-writes to PCM cells that require substantially higher programming due to PV. This proposed adaptive data comparison write (ADCW) scheme takes the advantage of asymmetric power characteristics on PCM reads and writes as well as the non-destructive PCM read. The ADCW performs a read operation before writes and only executes write operations on bits that are different from the previously stored data. Moreover, it uses OS page level memory access characteristics to dynamically enable/disable data comparison write operations. As shown in Figure 4-14(A), the implementation of ADCW involves a slight modification in the PCM peripheral circuitry and memory controller. For the memory controller, a new memory command \texttt{RD\_WD\_XOR} is added, which performs a bit-level comparison between the previously stored data (which is read out and stored in read latches) and the data to be written into the cells (which is stored in write FIFO). Therefore, two additional memory commands (i.e. read and \texttt{RD\_WD\_XOR}) will be generated upon a write memory transaction. The outcome of the XOR operation is used to update \texttt{Write\_Driver\_Enable\_Register}, which provides control for PCM peripheral circuits to enable/disable the write operations on a per-bit basis so that only the bits whose states differ from the input bits will be updated.
Note that another implementation of data comparison write is proposed in [75] to extend PCM lifespan. The proposed ADCW is different with [75] in that this scheme can significantly reduce DCW-induced performance overhead in typical PCM design scenarios. In fact, due to the increased write latency in DCW, applying DCW to every PCM write will degrade performance by up to 23%. [75] claims that the increased write latency does not have a significant impact on performance. The reasons for this discrepancy are twofold: (1) [75] assumes that the PCM peripheral circuit is similar to the one used for conventional DRAM, which is capable of reading/writing the entire cache line simultaneously at a cost of considerably increased die area. With this assumption, the increased delay due to DCW for writing back a dirty cache line is the same as the latency of a single-bit read. To reduce the die area occupied by the peripheral circuit and improve the effective capacity, the majority of recently announced PCM prototypes adopt design that has a reduced number of bit-line S/As, leading to a narrower width of parallel bit-read/write. For example, the prototype presented in [81] has 64 S/As per bank and only 64-bit read/write can be performed simultaneously. In this case, the increased delay due to DCW is 8X (64 × 8-bit / 64-bit = 8) of a bit-read latency for a 64-byte cache line write back. (2) Writes can be performance-critical when the utilization of the transaction queue is high. Increasing write latency is likely to increase the probability of this queue filling up during program executions that exhibit burst write access patterns. Consequently, the following read requests will be blocked due to the full queue and result in performance penalty. In this work, a transaction queue of 32 entries is modeled.

Instead of applying DCW to every PCM writes, an adaptation scheme is proposed that dynamically enables/disables DCW based on OS-level page classification. Since variations are layout dependent, how the PCM cells located at different regions are accessed by programs can
have a significant impact on the PV-induced overhead. For example, when programs write frequently to the memory regions that require higher write current than the nominal value, the PV-induced overhead will be exacerbated. This motivates us to use OS paging to map hot-modified pages (e.g. pages #1000, #1001, #1002 in Figure 4-14(B)) to PCM regions that are positively affected by PV and allocate PCM regions that are negatively affected by process variation to cold-modified pages in which data is infrequently updated. This proposed scheme dynamically applies DCW on writes to the cold-modified pages (as shown in Figure 4-14(A)), which allows us to minimize PV-induced overhead while yielding minimum impact on performance. This is because the low frequency writes are unlikely to fill up the transaction queue and block the following read requests. To identify hot- or cold- modified pages, the modification counters are used to track the frequency of page updates and employ the Multi Queue Algorithm [62] for page classification. Using circuit-level design tools and a modified DRAMsim simulator, it is estimated that the hardware-implemented algorithm and modification counters require a total die area of 0.35mm² and consume an approximate 11.1mW power. Our simulation results show that the additional power overhead induced by ADCW offsets less than 9% of power savings achieved by DCW.

**Adaptive Memory Compression with PCM Cell Refreshing**

As mentioned earlier, partial writes [74] reduce the number of bit-writes by eliminating the unnecessary write back of clean words and can be used to mitigate PV-induced overhead. Partial writes perform well when a majority of the words in a cache line are unmodified, and its benefit diminishes as the number of dirty words increases. As a worst case scenario, no benefit is achieved by partial writes when all words in the entire cache line are dirty. By profiling the number of dirty words in the evicted cache lines across all 18 benchmarks from various benchmark suites, it is observed that on average equal to or more than 15 out of 16 words are
modified in 62% of cache lines that are evicted from the last level cache (i.e. L2 cache with 64 Byte line size), as shown in Figure 4-15. This observation indicates that the worst case or near-worst case scenario occurs frequently.

To address PV-induced overhead on writes dominated by dirty words, a novel adaptive memory compression technique with cell refreshing is developed. This technique has two major components. The first component is OS-guided adaptive memory compression, which selectively compresses the evicted dirty cache lines, prior to writing them back to PCM. Figure 4-16 illustrates the scheme proposed. Similar to the adaptive DCW, the OS-level paging scheme is employed to perform page classification and remapping. In the memory controller, the current provision mechanism performs a lookup in the storage containing post-fabrication tuning information upon receiving an evicted cache line. If the voltage level required for writes is smaller than the average level, it implies a write-back of dirty cache lines to a region that is positively affected by PV and we apply default partial writes on these write-backs. Otherwise, the write-back will update a memory region that is negatively affected by PV. In such cases, the cache lines are selectively compressed if compression outperforms partial writes in terms of bit-writes reduction. Note that compression and partial writes are mutual exclusive schemes and they cannot be applied simultaneously, because compression changes the form of data in memory and thereby requires writing back the entire compressed cache line.

To exploit adaptive compression, a table, Block_Compression_Table, is used, which assigns one bit for each cache line to indicate the status of data block in main memory (e.g. “1” : compressed, “0” : uncompressed). Then a lookup in this table is performed and the result is loaded to the Block_Status register. In parallel with Block_Compression_Table lookup, it is determined that whether the compression should be invoked by monitoring the number of dirty
words present in the evicted cache line. The number of dirty words is tracked by a fine-grained dirty bit proposed in partial writes [74]. One important design parameter is the threshold of the number of dirty words that will trigger compression. Compression can either greatly reduce or considerably increase PV-induced overhead, depending on two factors: the number of dirty words and data compressibility. As compressibility can only be obtained after compression, the number of dirty words is used as an indicator to trigger the compression and update memory with compressed data only when it is smaller than the original size. A sensitivity analysis is performed and it is observed that choosing a threshold of 14 (e.g. Compression_Threshold = 14) provides the maximum benefit across all simulated workloads. When the number of dirty words is smaller than this threshold, compression is not invoked. Then only dirty words are written back if the cache line is in uncompressed form in memory (i.e. Block_Status = 0), or the entire cache line (in uncompressed form) is written back if the data in memory is in compressed form (i.e. Block_Status = 1). When the number of dirty words reaches the threshold, compression is invoked. If the compressed cache line has a larger size then the uncompressed one due to extra bits required by prefix, the uncompressed data are used for updating memory. The Block_Status register is updated and its data is written back to Block_Compression_Table. Otherwise, compressed data is written back to memory.

As compression is not used for improving effective memory capacity in this work, the same amount of memory space (64Byte in this case) is used to hold compressed or uncompressed data. Instead of writing compressed data starting at a cache line aligned boundary, a simple line shifting mechanism is applied to even out the writes within a PCM block, as shown in Figure 4-17. The shifting is performed on a byte granularity and the first byte is reserved for storing Head_Index (highlighted in dark grey), which records the offset of the first byte. Each
update on the memory block will result in shifting the compressed data right by one byte and _Head_Index_ is incremented by one, as shown in Figure 4-17. The byte shifted off the right end wraps around to the second byte at the left end (byte “2B” in Figure 4-17(C)). The power savings may be offset by the power overhead of the compression procedure. It is estimated that the overall power overhead of compression/decompression only accounts for 17% of power saved due to compression mechanism. Moreover, a noticeable performance improvement is observed, because the reduced number of bit-read/write due to compression can decrease the number of PCM accesses, whose latency (i.e. 250ns) is 600X higher than compression/decompression delay (i.e. 0.41ns).

The second component of the proposed technique is a new programming approach that can refresh long-cycled PCM cells. A recent work [93] which presents experimental evidences shows that reversing the programming current direction can refresh a degraded, long-cycled PCM cell. The physical mechanism behind reverse programming is that field-induced atomic migration is a principle mechanism of endurance degradation. By reversing programming current, the direction of migration is reversed and therefore the endurance is extended. To identify long-cycled PCM cells, built-in ECC is used as an indicator to locate memory blocks that contain degraded cells. In this work, it is assumed that an off-chip memory bus with a width of 64-bit (8-byte) and a single-error correction code, Harming-based ECC, are used. Upon a bit-error detected and corrected by ECC, the 8-byte aligned physical address that contains the error bit is inserted into a queue, _refresh_queue_. Before each write-back of dirty cache line, the _refresh_queue_ is scanned to discover whether any address stored in the queue is covered by the write-back. If an address is found, reverse programming will be applied on the 8 bytes starting from the matched memory address. Then this address is removed from _refresh_queue_. To perform reverse programming, the
memory controller sends a reverse-programming signal to the PCM peripheral circuits to switch the input voltage of the charge pump from a positive voltage to a negative one, resulting in write current flowing in the opposite direction. After that, the entire cache line is written back to memory since reverse programming destroys the data previously stored in memory.

To implement the proposed scheme, a DRAM-based storage is used to implement Block_Compression_Table and a hardware-implemented Frequent Pattern Compression (FPC) [94] design is employed. For a 2GB PCM, Block_Compression_Table requires a 4MB storage capacity with an area overhead of 0.2% of total die area and an additional power consumption of 8.4mW. The FPC can compress and decompress a cache line in 0.41ns [95]. The area overhead and dynamic power consumption of compressor and decompressor modules at 45nm technology are 0.183mm² and 0.273W respectively [95]. Our simulation results show an average compression ratio (i.e. compressed size/uncompressed size) between 0.47-0.98 across all simulated benchmarks, as shown in Table 4-3. For the reverse programming technique, it is estimated that an area overhead of 1% with a 0.02mW power overhead and additional 2ns delay. All of the power and performance overhead has been taken into account in the simulation results.

**Experimental Methodology**

In this section, the experimental methodology for evaluating the benefits of the proposed variation-aware PCM design is described. A quad-core system comprised of four out-of-order processors with the parameters given in Table 4-1 and a PV-affected off-chip 2GB PCM memory are simulated. It is assumed that a 45nm technology with a supply voltage of 1.1V is used. Monte-Carlo simulation is used in the variation analysis, which takes into account both die-to-die and within-die variations. A 3-level quad tree method is used to model variation correlations related to layout geometrics. 400 PCM chips are generated and each chip has a unique write current profile obtained through Monte-Carlo simulation and one-dimensional heat model
introduced previously. Then a number of 400 architecture simulations are performed to capture
die-to-die variation. The characterization of PV impact on power and endurance is performed for
each individual PCM chip. The results are reported as an average value or statistical distribution.
To evaluate proposed techniques, a diverse set of memory intensive applications are used from
various suites to cover a wide range of compressibility properties, miss rates and working set
size. These applications are listed in Table 4-3 along with their average data compression ratio
and memory reference characteristics including memory footprint size and Miss Per-Kilo
Instruction (MPKI), when they run in standalone mode on a single core with a 2MB L2 Cache.
Six programs (mcf, art, apsi, lucas, gzip and gap) that exhibit the highest MPKI among all
SPEC2000 benchmarks with reference inputs are selected. Eight programs (IS, UA, BT, CG, FT,
LU, MG and SP) from NAS Parallel Benchmarks Version 3.2 with Class “C” input data set are
used, featuring a gigabyte working set. Two programs (utility_mining and kmean) are chosen
from MineBench [96] with real-world data as input datasets. Qsort is a Unix utility and Triad is a
streaming benchmark derived from the STREAM suite[97]. A random number generator is used
to generate a sequence of 50M integers as inputs to these two benchmarks. All benchmarks are
compiled on an x86 platform using GCC or FORTRAN compiler with optimization level –O3.
To form 4-threaded multiprogramming workloads, we first categorize all benchmarks into: high-
miss (MPKI>190), moderate-miss (190<MPKI<60), low-miss (MPKI<60) groups. In Table 5-4,
the High-, Moderate- and Low- miss workloads (H1-H3, M1-M3, L1-L3) consist of four
benchmarks exclusively from each category. The High-Moderate- and Moderate-Low miss
workloads (HM1-HM3, ML1-ML3) are formed by using two benchmarks from each category.
For all experiments, SimPoint toolset is used to choose representative execution intervals for all
benchmarks and perform detailed cycle-level simulation until at least 1 Billion instructions are committed on every core.

For performance and power evaluation, a framework based on a heavily extended full-system simulator, PTLSim/X[98], integrated with a modified memory model, DRAMSim[60] is developed. PTLSim/X is a cycle accurate simulator supporting the x86 instruction set architecture. PTLSim/X is extended to model a 2-level write back cache, contention for memory buses, bus traffic, memory controller request queue (including a 4-entry write buffer to optimize write operation to PCM) and memory compression. To model the latency and energy of PCM memory, the DRAMsim timing module is enhanced to model PCM-specific peripheral structures (current sense amplifier and write driver blocks) and extended its power module for PCM energy estimation. For the timing and power parameters, the range of values for a given PCM parameter is found through an extensive literature search and the median value for that parameter is used. The parameter values in this work are listed in Table 4-2 and they are similar to the ITRS 2007 projection for PCM technologies [69]. The variations in RESET/SET currents are obtained through a developed analytical model and SPICE simulation using Predictive Technology Modes, known as BSIM[43] for a 45 nm technology. A disk access latency of 4.2ms [63] on an IDE disk is assumed. The harmonic IPC of quad-core processor is used as our performance metric. To implement the OS paging scheme, the custom memory manager in PTLSim is modified to support page migrations. This modified memory manager is responsible for maintaining TLB coherence, copying the page to its new home (this migration is evaluated by invoking a bcopy() routine) and flashing the cache lines belonging to the pages to be migrated. The associated performance and power overhead have been taken into account in our reported simulation results.
Results

In this section, the power and endurance benefit of using the proposed variation-aware PCM design techniques are presented. All results reported are computed as an average over the 400 simulated PCM chips. The experimental evaluation includes: 1) power savings achieved by variation-aware schemes and their performance overhead; 2) life span extension by adopting variation-aware techniques; 3) sensitivity analysis of tuning resolution.

Power Savings

Figure 4-18 shows the average power savings benefit of the proposed PV-aware PCM optimization techniques. The bar labeled “gmean” denotes the geometric mean. The “NoPV” case, which is a PCM without process variations, is used as the baseline. All other results are normalized with respect to the “NoPV” case. The “PV” represents the PCM implementation that conservatively tolerates process variations based on the worst case scenario. The “CP” scheme uses the proposed process-variation aware Current Provision (CP) technique to adapt programming current. “CP+PW” is a design that combines the Current Provision with Partial Writes (PW). This “CP+PW” design can be further coupled with OS-guided DCW and OS-guided Adaptive Memory Compression (AMC) to form “CP+PW+ADCW” and “CP+PW+AMC”. The last bar, “CP+PW+ADCW+AMC” represents the case where all of the proposed techniques are applied simultaneously. As shown in Figure 5-18, the worst-case-scenario-driven PCM increases write power by 96% in geometric mean. While this power overhead can be significantly reduced to 10% by PV-aware current provision. With partial writes employed, “CP+PW” achieves an additional 13% write power savings over “CP”. This additional benefit is primarily contributed by benchmarks sort, FT, Utility, mcf and art, which have a small number of dirty words in the evicted dirty cache lines as shown in Figure 4-15. Moreover, it is found that a 22% and 17% write power savings can be achieved by
“CP+PW+ADCW” and “CP+PW+AMC” respectively because of the fewer bit-writes on memory regions that are negatively affected by PV. Note that all mix workloads benefit more from OS-guided DCW than OS-guided adaptive memory compression, except for low miss workloads. This is because statistically DCW can remove bit-writes by 50% due to the equal possibility of writing a “0” and “1”, while compression is not as effective as DCW in bit-write reduction because of an average 74% compression ratio observed across our simulated benchmarks. For Low-Miss workloads, DCW and Compression are not invoked frequently. This is due to the fact that the total dataset resident in main memory for Low-Miss workloads is far less than the total memory capacity and therefore OS-paging can effectively re-map hot-modified pages to free memory regions that are positively affected by PV, while other workloads require the full space of main memory and limit the opportunities for page remapping. DCW can provide a larger power savings than adaptive compression, but at the expense of substantial performance degradation due to extra memory read accesses. Figure 4-19 shows IPC values for the different techniques. DCW incurs a 7% harmonic IPC penalty, while adaptive memory compression improves performance by 16%. Among all techniques, “CP+PW+ADCW+AMC” achieves the best tradeoff between power reduction and performance degradation. It achieves 63% power savings and 11% performance improvement that is similar to “CP+PW” over the “PV” case.

Figure 4-20 presents the normalized power consumption distribution in the presence of PV across the simulated 400 PCM chips as well as a scenario after applying all proposed PV-aware optimization scheme. A power value of 1 corresponds to the power consumption without PV effect. Of PCM chips not using PV optimization schemes, all of the chips exhibit power consumption greater than that of an ideal design, while 31% of the chip exhibit equal or greater than 2X power consumption. In contrast, with the optimization schemes, 9% of the chips
consume power that is greater than an ideal design. No chip exhibits greater than 2X power consumption.

**Endurance Enhancement**

To estimate wear-out induced failures, each workload is executed repeatedly and the number of writes to each bit is tracked. $1 \times 10^5$ instead of $1 \times 10^8$ writes is used in the lifetime estimation due to the extremely long simulation time to collect $1 \times 10^8$ writes to any PCM cell. With this accelerated estimation method, when a bit has more than $1 \times 10^5$ writes on it, it is marked as a failed cell. A memory access a failure is defined as when the number of failed bits accessed in a memory reference is larger than the number of errors that can be corrected (it is assumed that ECC with single error detection/correction is employed). The lifetime of PCM-based memory is estimated as the number cycles elapsed before the first memory access failure occurs. Figure 4-21 shows the lifetime improvement achieved by our proposed techniques. All results are normalized to the lifetime of the “NoPV” case. Compared to the “NoPV” case, the endurance of PV-affected memory is substantially shortened by a factor of 50X because of the considerably large magnitude of current applied to program memory cells. By adapting the current used for writes, “CP” can significantly increase the endurance by 27X with respect to the “PV” case. Furthermore, “CP+PW” extends the lifetime by another 7X in average, relative to “CP”. The “CP+PW+ADCW” and “CP+PW+AMC” allow a geometric mean of 277X and 268X boost in endurance respectively over the “PV” case and they make the lifespan of PCM memory system 5.5X and 5.3X better than that of “NoPV” case. Note that the lifetime extension for high miss workloads is larger than that for low miss workloads. This is because the applications in the high- and moderate- miss categories stress memory more intensively and also exhibit a large memory footprint than others, leading to a shorter lifetime on the baseline design. By combing all techniques together, “CP+PW+ADCW+AMC” has the ability of achieving 13050X (with cell
refreshing) and 488X (without cell refreshing) endurance respectively over the “PV” case. Note that we assume the refresh technique is able to extend a cell lifetime by 35 times, which has been experimentally demonstrated in [93].

**Sensitivity Analysis of Tuning Resolution**

Although our post-fabrication tuning allows using fine-grained current control to mitigate the PV-induced overhead, this tuning optimization is sensitive to the selection of tuning resolution. Moreover, the die area overhead and access latency required for storing and retrieving post-fabrication information should be considered. Due to the spatial effect of PV, using a very fine-grained tuning resolution may not be able to provide significant benefit, but incur more area and complexity overhead. A sensitivity analysis is performed and Figure 4-22 shows the power and endurance benefit of “CP” as the tuning resolution increases from one bit to one memory region of 40MB in size. Note that the results are normalized to the per-bit level tuning. It is found that the benefits decrease dramatically when the tuning resolution drops below 4MB. The per-bit level tuning achieves only about 15% more power savings and 24% longer lifespan, while the die area overhead is 100% and the performance degradation is about 14% due to the longer latency in fetching post-fabrication data from a large flash storage. Thus a tuning resolution of 4MB is selected that provides a good trade-off between power/endurance benefit and area/complexity overhead.

**Related Work**

As process variation significantly affects performance and leakage power consumption of microprocessor, there have been many techniques proposed in literature to mitigate the effect of process variation. For example, Body Biasing is widely applied to mitigate PV effect[83] and make a tradeoff between performance and power[85]. [86] proposes cycle time stealing on pipeline stages for frequency improvement. [87] adopts a novel 3T1D DRAM to implement
memory for tolerating PV. [88] applies voltage interpolation and variable latency tuning techniques. However, existing work on analyzing and mitigating the impact of process variation has been largely focus on CMOS devices and circuits. This work studies the impact of PV on the novel phase change memory design, which is based on a new emerging memory technology.

Due to superior scalability and low power features, phase change memory is attracting increasing attention as one of the most promising technologies for next generation memory. To our knowledge, [74][75][99][100] are among the first architecture level studies on using PCM to implement main memory. [74] examines PCM buffer organization and propose partial writes to tolerate long latency and high energy of PCM writes. [99] explores a hybrid PCM/DRAM memory with the latency benefits of DRAM and capacity benefit of PCM. [75] presents a set of techniques to extend the lifetime of PCM-based memory, such as redundant bit-write removal, row shifting and segment swapping. This study differs from these works in two ways: 1) this work focus on process variation of PCM, while their work doesn’t take variability into consideration. 2) the proposed techniques in this dissertation address process variation issues at the circuit-, microarchitecture- and OS- level, whereas their schemes are built only on the architecture layers.
Table 4-1. Baseline machine configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>3GHz</td>
</tr>
<tr>
<td>Width</td>
<td>4-wide fetch/decode/issue/commit</td>
</tr>
<tr>
<td>IQ</td>
<td>64 entries</td>
</tr>
<tr>
<td>ITLB</td>
<td>128 entries, 4-way</td>
</tr>
<tr>
<td>BranchPred</td>
<td>2K entries Gshare, 10-bit global history</td>
</tr>
<tr>
<td>BTB</td>
<td>2K entries, 4-way</td>
</tr>
<tr>
<td>RAS</td>
<td>32 entries RAS</td>
</tr>
<tr>
<td>L1 ICache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle</td>
</tr>
<tr>
<td>ROB Size</td>
<td>128 entries</td>
</tr>
<tr>
<td>LDQ</td>
<td>48 entries</td>
</tr>
<tr>
<td>STQ</td>
<td>32 entries</td>
</tr>
<tr>
<td>Int ALU</td>
<td>4 I-ALU, 2 I-MUL/DIV, 1 Load/Store</td>
</tr>
<tr>
<td>FP ALU</td>
<td>2 FP-ALU, 2 FP-MUL/DIV/SQRT</td>
</tr>
<tr>
<td>DTLB</td>
<td>256 entries, 4-way</td>
</tr>
<tr>
<td>L1 D-Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>shared 2MB, 8-way, 64 Byte/line, 12 cycle</td>
</tr>
<tr>
<td>Memory</td>
<td>PCM, 2GB, 8 banks</td>
</tr>
<tr>
<td>Write Buffer</td>
<td>4 entries, 512B per entry</td>
</tr>
</tbody>
</table>

Table 4-2. PCM timing and power parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>tRCD</td>
<td>60</td>
</tr>
<tr>
<td>tCAS</td>
<td>12</td>
</tr>
<tr>
<td>tWR</td>
<td>55/250 for RESET/SET</td>
</tr>
<tr>
<td>tCMD</td>
<td>6</td>
</tr>
<tr>
<td>RESET current</td>
<td>0.3mA</td>
</tr>
<tr>
<td>SET current</td>
<td>0.2mA</td>
</tr>
</tbody>
</table>

Table 4-3. Benchmarks used to form workloads

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>MPKI (2MB L2)</th>
<th>Memory Footprint (MB)</th>
<th>Average Compression Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAS</td>
<td>CG</td>
<td>441</td>
<td>914</td>
</tr>
<tr>
<td></td>
<td>BT</td>
<td>252</td>
<td>691</td>
</tr>
<tr>
<td></td>
<td>SP</td>
<td>241</td>
<td>726</td>
</tr>
<tr>
<td></td>
<td>MG</td>
<td>214</td>
<td>432</td>
</tr>
<tr>
<td></td>
<td>LU</td>
<td>176</td>
<td>601</td>
</tr>
<tr>
<td></td>
<td>IS</td>
<td>192</td>
<td>1056</td>
</tr>
<tr>
<td></td>
<td>UA</td>
<td>123</td>
<td>486</td>
</tr>
<tr>
<td></td>
<td>FT</td>
<td>120</td>
<td>1284</td>
</tr>
<tr>
<td>Mine</td>
<td>utility_mining</td>
<td>93</td>
<td>547</td>
</tr>
<tr>
<td>Bench</td>
<td>kmean</td>
<td>78</td>
<td>402</td>
</tr>
<tr>
<td>Unix utility</td>
<td>qsort</td>
<td>75</td>
<td>722</td>
</tr>
<tr>
<td>STREAM</td>
<td>triad</td>
<td>67</td>
<td>938</td>
</tr>
</tbody>
</table>
### Table 4-3. Continued

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>MPKI (2MB L2)</th>
<th>Memory Footprint (MB)</th>
<th>Average Compression Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>mcf</td>
<td>37</td>
<td>98</td>
<td>0.47</td>
</tr>
<tr>
<td>apsi</td>
<td>30</td>
<td>196</td>
<td>0.63</td>
</tr>
<tr>
<td>art</td>
<td>30</td>
<td>4</td>
<td>0.65</td>
</tr>
<tr>
<td>lucas</td>
<td>21</td>
<td>91</td>
<td>0.88</td>
</tr>
<tr>
<td>gzip</td>
<td>17</td>
<td>75</td>
<td>0.90</td>
</tr>
<tr>
<td>gap</td>
<td>14</td>
<td>192</td>
<td>0.62</td>
</tr>
</tbody>
</table>

### Table 4-4. Workloads

<table>
<thead>
<tr>
<th>Type</th>
<th>Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>H1(CG,BT,SP,MG)</td>
</tr>
<tr>
<td></td>
<td>H2(SP,MG,LU,IS)</td>
</tr>
<tr>
<td></td>
<td>H3(CG,BT,LU,IS)</td>
</tr>
<tr>
<td></td>
<td>HM1(CG,BT,Sort,Triad)</td>
</tr>
<tr>
<td>High &amp; Moderate</td>
<td>HM2(SP,MG,Utility,kmean)</td>
</tr>
<tr>
<td></td>
<td>HM3(LU,IS,UA,FT)</td>
</tr>
<tr>
<td></td>
<td>M1(UA,FT,Utility,kmean)</td>
</tr>
<tr>
<td>Moderate</td>
<td>M2(Utility,kmean,sort,triad)</td>
</tr>
<tr>
<td></td>
<td>M3(UA,FT,Sort,Triad)</td>
</tr>
<tr>
<td></td>
<td>ML1(UA,FT,gzip,gap)</td>
</tr>
<tr>
<td>Moderate &amp; Low</td>
<td>ML2(Utility,kmean,art,lucas)</td>
</tr>
<tr>
<td></td>
<td>ML3(Sort,Triad,mcf,apsi)</td>
</tr>
<tr>
<td>Low</td>
<td>L1(mcf, apsi,art,lucas)</td>
</tr>
<tr>
<td></td>
<td>L2(art,lucas,gzip,gap)</td>
</tr>
<tr>
<td></td>
<td>L3(mcf,apsi,gzip,gap)</td>
</tr>
</tbody>
</table>
Figure 4-5. The electrical paths of PCM write operations.

Figure 4-6. Variation in the width of $I_{\text{RESET}_\text{min}}$ due to transistors’ length variation

Figure 4-7. Variation in the amplitude of $I_{\text{RESET}}$ delivered to PCM cell due to transistors’ length variation.

Figure 4-8. Variation in the amplitude of $I_{\text{RESET}_\text{min}}$ due to BECD variation
Figure 4-9. Variation in the amplitude of $I_{\text{RESET},\min}$ due to ThickHEATER variation.

Figure 4-10. Variation in the amplitude of $I_{\text{RESET},\min}$ due to ThickGST variation.

Figure 4-11. The color map and distribution of minimum required programming current for a PCM chip
Figure 4-12. The cycle lifetime as a function of programming pulse energy [24].

Figure 4-13. Fine-grained voltage tuning for write driver circuits.

Figure 4-14. (A) The implementation of adaptive data comparison write (ADCW) and (B) OS-level page classification.
Figure 4-15. A profiling of the number of dirty words in the evicted cache lines.

![Profiling of Dirty Words](image1)

Figure 4-16. Adaptive memory compression.

![Adaptive Memory Compression](image2)

Figure 4-17. Line shifting mechanism.

![Line Shifting](image3)

Figure 4-18. A comparison of write power saving.

![Write Power Saving](image4)
Figure 4-19. The performance impact of variation-aware PCM design.

Figure 4-20. Power consumption distribution.

Figure 4-21. A comparison of endurance improvement.

Figure 4-22. Sensitive analysis of tuning resolution.
CHAPTER 5
A RESISTANCE DRIFT RESILIENT ARCHITECTURE FOR MULTI-LEVEL CELL PHASE CHANGE MEMORY SYSTEM

PCM relies on the phase change properties of a chalcogenide material, typically Ge$_2$Sb$_2$Te$_5$ (GST), to represent stored information. A PCM cell can be reversibly switched between two structural phases (i.e. amorphous and crystalline states) with significantly different resistances (i.e. high in the amorphous and low in the crystalline). While the crystalline phase is fairly stable versus time and temperature, the amorphous phase is meta-stable and can experience amorphous state resistance drift [101]. The increase in resistance for the amorphous state represents a concern for PCM reliability, for which the resistance value is indicative of the logic state, and thus must be constant over time. For single-level programming PCM device, there are 2-3 orders of magnitude resistance difference between set (i.e. crystalline) and reset (i.e. amorphous) states. Therefore, the impact of resistance drift is negligible and the drift may even be beneficial for distinguishing one state from the other in single-level cells.

However, the continued quest for high integration density has recently motivated the PCM R&D community to design and fabricate PCM devices with multi-level cells (MLC) [102][103][104]. To store more bits in a single PCM cell, the MLC-PCM relies on fine-grained resistance partitioning, leading to a reduced resistance margin that separates the states. Consequently, drift induced resistance increase is more likely to result in unwanted mixing of intermediate states and more errors for MLC-PCM. To address this issue, conventional MLC-PCM design adopts a wide resistance margin between any adjacent states to guard against drift. This large resistance margin causes the resistance window bounded by the lowest and highest resistance states to be enlarged. As the lowest resistance states exhibit a fixed resistance property, the resistance of the highest resistance states needs to be increased. As a result, an increased programming current is required to program PCM devices, resulting in high power...
consumption and degraded endurance. In this work, a first step is made in quantifying the impact of runtime program execution and chip temperature on MLC-PCM resistance drift. Then a novel architecture, Helmet, is proposed which is able to cost-effectively reduce drift-induced read out error rate. Alternatively, Helmet can relax the margin size requirement while achieving the same reliability target as the conservative approach. Consequently, the proposed design is capable of alleviating the deleterious impact of wider margin size on MLC-PCM power and endurance.

**Background: Multi-level Cell Phase Change Memory (MLC-PCM) and Resistance Drift**

**Multi-level Cell Phase Change Memory**

Phase Change Memory is a class of non-volatile memory that exploits the property of the phase change material (GST) to switch between amorphous and crystalline states with the application of electrical-pulse generated heat. PCM devices are typically programmed in the single level cell (SLC) mode, namely, the phase change material is programmed either to the amorphous, high resistance state to represent a binary “0” (RESET) or the crystalline, low resistance state to represent a “1” (SET). Typically, there are 2-3 orders of magnitude resistance difference between the SET and RESET state [101], which provides the opportunity to represent more states within a cell. This advantage has recently been exploited by a number of semiconductor manufacturers [102][103][104] to design and prototype MLC-PCM due to its high density and low per-bit fabrication cost. Instead of using two resistance levels, the MLC-PCM devices can be programmed to one of the multiple resistance levels (i.e. intermediate states) between the pre-defined minimal and maximum resistance. In the intermediate states, the active region of GST material is partially amorphized and its fraction needs to be precisely controlled so that the desired resistance state can be obtained with sufficient accuracy. Therefore, MLC-PCM requires higher programming precision than its SLC counterpart. To achieve this, MLC-PCM normally adopts a multilevel programming algorithm [104][105] shown in Figure 5-
1. This programming algorithm is based on a program-and-verify (PAV) technique. The cell is first programmed to its lowest resistance state by using a long SET pulse, followed by a single RESET pulse to amorphize the entire active region of the phase change material. This initialization sequence prepares the cell for the subsequent stair-case up (SCU) sequence, which switches a fraction of the active region of the GST material from amorphous state to crystalline state, depending on the target state. During the SCU procedure, a sequence of pulses is applied on the programmed cell with a verify step following each pulse. All pulses have the same width, but each pulse has an increased magnitude over the preceding one. As the crystalline fraction of phase change material grows with the elevated temperature and the elapsed time, the increased number of pluses and the raised current amplitude will switch an increased fraction of GST to crystalline state from amorphous state. This SCU procedure stops once the desired resistance level is detected at the verify step. Although the aforementioned programming algorithm achieves a great precision in programming, the resulting resistance value for a logic state still varies across memory cells due to variations in fabrication processes. Therefore, for a multi-level cell, each logic state is represented by a range of resistance values rather than a unique number and a sufficient resistance margin between any two adjacent states is required to unambiguously distinguish one state from another. This resistance margin depends on the sensing current margin required by sensing circuits, since data stored in PCM cells are retrieved by (1) applying a predefined bias voltage across the cell and (2) sensing the generated current amplitude to determine the resistance states.

**Resistance Drift and its Impact on MLC**

After a PCM cell is programmed, its resistance value can increase with time and the increment saturates over time. This phenomenon is known as resistance drift. Resistance drift is believed to be the result of structural relaxation (SR) physical phenomena, which is a thermally-
activated, atomic rearrangement of the amorphous structure [106]. It has been observed
[104][107] that drift is much more significant on high resistance states (e.g. “00”, “01”, “10”), in
which a volume of the phase change material is programmed to the amorphous states, such as the
fully amorphous state (e.g. “00”) or the partially amorphous states (e.g. “01”, “10”) in the MLCs,
whereas the low resistance state (e.g. “11”) shows a nearly negligible dependence of resistance
on time. The increased rate of resistance exhibits a power-law behavior with the time elapsed
after programming, which can be described as:

\[ R = R_0 \left( \frac{t}{t_0} \right)^\nu \]  

(Eq 5-1)

where \( R_0 \) is the resistance at time \( t_0 \), and \( \nu \) is the drift coefficient. The drift coefficient
has a wide range from 0 to 0.12 [104][107], depending on the resistance level and temperature.
At a given temperature, the drift coefficient increases monotonically with the resistance value
following a logarithmic law [107]. On the other hand, for a given programmed resistance level,
resistance drift is accelerated by temperature since \( \nu \) increases with the elevated temperature
[108]. Note that Eq 5-1 is a saturation function since the exponent, \( \nu \), is smaller than 1. It
suggests that resistance drift is very fast at the beginning and saturates with time, as shown in
Figure 5-2(A).

Resistance drift normally doesn’t cause problems in conventional single-level PCM cells.
This is because the resistance of the amorphous state slowly increases while crystalline state
remains at a stable resistance. Therefore, the resistance difference between the SET and RESET
becomes larger, thereby increasing the noise margin. However, this resistance drift represents a
reliability concern for MLC because of the much tighter margin in MLCs. Figure 5-2(B)
illustrates a case in which there is nearly no margin between adjacent states and shows the
resistance distribution before and after drift for an elapse of 1 minute. Note that the resistance

distribution is wider for the high-resistance state (e.g. “00”) due to the variation in the volume size of amorphous regions after programming. As can be seen in Figure 5-2(B), resistance drift can significantly decrease the reliability of data retrieval by shifting the resistance of a programmed cell out of its resistance range and cause overlap in adjacent states. As shown by the shadow area in Figure 5-2(B), a memory cell that was initially programmed into the intermediate state (e.g. “01”) may transit to a higher-resistance state (e.g. “00”) after 1 minute if there is no margin between the two states. Subsequently, the memory cell may be read as being in an erroneous state by the sense amplifier, resulting in a data corruption. In contrast to state “01”, state “10” is less susceptible to drift. This is because the resistance value of “10” is smaller than that of “01” and therefore has a lower drift coefficient. On the contrary, resistance drift will not cause either state “11” or state “00” to be corrupted, since the former is a stable state and the resistance distribution of the latter will not overlap with other resistance partitions after drift.

One approach to tolerate drift in MLC is to increase the resistance margin between any adjacent states to prevent the post-drift resistance partitions from overlapping with pre-drift resistance partitions, as is illustrated in Figure 5-2(C). For example, there is approximately 5X resistance difference between any adjacent states (i.e. $R_{\text{state00}}/ R_{\text{state01}}= R_{\text{state01}}/ R_{\text{state10}}= R_{\text{state10}}/ R_{\text{state11}}=5$) in some recent published MLC-MPC prototypes [102][104]. By using Eq 5-1, it is estimated that 5X margin size allows data to be valid for 2 years at room temperature. To achieve this, the resistance window bounded by the lowest resistance state and the highest resistance state needs to be enlarged to make room for the increased margin. Doing so improves the resilience to drift-induced readout error, but at the cost of consuming more programming power and degraded cell endurance due to exposing all PCM cells to a higher programming current. On the contrary, if a smaller margin size, such as 2X (i.e. $R_{\text{state00}}/ R_{\text{state01}}= R_{\text{state01}}/ R_{\text{state10}}= R_{\text{state10}}/ R_{\text{state11}}=2$) is
employed, the data in MLC become invalid after only 1 hour at room temperature. At a high
temperature environment, the drift-induced resistance increase is more significant.

**Resistance Drift on MLC-PCM System: Characterization and Implication**

In this section, characterization of the dynamics of resistance drift in a MLC-PCM system
is presented and the impact of using conventional resistance drift tolerance techniques on PCM
power and endurance are discussed.

**A Characterization of Resistance Drift Dynamics on MLC-PCM System**

An analytical model is developed and integrated into the microarchitecture simulator to
quantify the readout error rate of MLC-PCM due to resistance drift. This analytical model is built
based upon Eq 5-1 to capture the behavior of resistance drift by tracking the dynamic resistance
change for each memory cell. It takes the following factors as inputs: the initial resistance value
for a logic level immediately after programming, memory reference characteristics and run-time
chip temperature. The simulated memory system consists of a large MLC-PCM serving as main
storage and a small DRAM between MLC-PCM and processor serving as a buffer. Previous
work[100][109][110] show that such hybrid memory architecture is able to combine the latency
benefit of DRAM and the low-power, high-density benefit of PCM. It is assumed that a 2-bit-
per-cell, 512MB(2GB effective capacity) PCM storage and a 64MB DRAM are employed and
both are managed at the granularity of a physical page size of 4KB. For the initial resistance
values used to represent each logic state, a range of design choices are evaluated that provide
different resistance margins. By doing so, it is able to quantify how the impact of drift varies
with the margin size. As described in the previous section, the resistance margin depends on the
read sensing current margin. Five cases are investigated, in which the read current margin is
increased from 0μA to 2μA with a step of 0.5μA and a larger current margin corresponds to the
case in which a wider resistance margin is employed and vice versa. In each case, the resistance
value assigned to each logic state is listed in Table 5-2. Each logic state is represented by a range of resistance values due to process variation. The multi-level quad tree approach [111] is used to model the correlated within-die and die-to-die variations in the resistance of each logic state across MLCs. It is assumed that the resistance distribution for each logic state follows a normal distribution with a coefficient variation ($\sigma/\mu$) of 17% which is similar to the parameters in recent prototypes [103]. Memory references to the PCM storage are collected using a cycle-accurate simulator and then feed into the analytical model, which tracks the amount of erroneous data that is read from the MLC-PCM storage. The data in a cell is considered as corrupted during read operations if the cell’s resistance value runs into its adjacent resistance range. Note that it is assumed that data can still be retrieved correctly even if the resistance value of the cell falls in the margin between adjacent states, such as the case shown in Figure 5-2(C). Therefore, the error-rate estimation is conservative and provides a lower bound estimation of error rates. A total of 27 benchmarks from the NAS Parallel benchmarks and SPEC2000 suite are simulated.

Figure 5-3 shows the quantitative results on the impact of resistance drift on readout error rate for a low-temperature off-chip MLC-PCM-based hybrid design and a 3D die-stacked on-chip design. It is assumed that the 3D on-chip design only differs from the off-chip design in terms of operating temperature for examining the impact of the elevated temperature on the drift-induced reliability issue. In the figure, the benchmarks are ordered in descending order of their drift-induced error rate along the X-axis and the absolute error rate (i.e. the number of erroneous bits divided by the total number of bits read from the PCM storage) is shown on the Y-axis. Five different resistance margins are investigated and each margin setting corresponds to a different resistance range bounded by the lowest-resistance state and the highest-resistance state.
As can be seen in Figure 5-3, there is a high probability of erroneous readout when the margin is too narrow (i.e. 0uA or 0.5uA margin), since drift can cause the resistance level to shift across the margin area from a correct to an incorrect state. With an increased margin setting (i.e. 1.5uA or 2uA margin), the likelihood of drift-induced corruption is mitigated, resulting in a one or two magnitude order reduction in the error rate. Therefore, a wider margin provides more tolerance for resistance drift and improves the read reliability. Moreover, it is observed that the simulated benchmarks exhibit a variety of read reliability characteristics as the resistance margin changes. This is because the PCM drift dynamics are also affected by memory reference and data patterns, both of which are workload inherent. For the former, the memory reference read-to-write latency, which is the measurement of the elapsed time (i.e. the number of cycles) between read references to a given page on the PCM partition and the most recent update to the same page, strongly affects the drift-induced read reliability since the initial resistance values can be restored upon on a write operation. As drift causes resistance increases over time, the higher the read-to-write latency is, the more likely an erroneous readout will occur. For example, on a given margin (e.g. 0.5uA) for the off-chip memory design, gzip exhibits an error rate that is 4X as high as that of swim. By analyzing the memory reference traces, it is observed that the average read-to-write latency on PCM partition for gzip is 95X greater than that for swim. As described in the previous section, resistances drift differently on different logic levels. Intermediate states, especially those with high resistance values (such as “01” shown in Figure 5-2(A)), are much more susceptible to the drifting phenomena. The value pattern, which varies with workloads and input, contributes differently to readout errors across benchmarks. For example, the analysis showed that both lucas and mesa share a similar average read-to-write latencies. However, lucas exhibits a significantly higher error rate than mesa. For a given margin (e.g. 0.5uA) in the off-
chip design, the drift-induced error rate for *lucas* is 0.9%, which is about 5X as high as that for *mesa*. This is because 52% of PCM cells referenced by read operations in *lucas* are in the intermediate states, namely “01” or “10”. On the contrary, *mesa* shows a pattern in which most cells contain either “00” or “11” during read operations and only 14.2% of cell reads from memory are in the intermediate states. Besides time and data pattern factors, resistance drift also shows significant temperature dependence. Compared to the low-temperature off-chip design, the elevated temperature in the 3D die-stacked design causes an up to 2X increase in the error rate, as is shown in Figure 5-3. The higher error rate is contributed by the increased drift due to a higher drift coefficient. Hence, an even wider margin is required to tolerate drift to ensure a low bit error rate in light of elevated chip temperature.

**Implication of Conservative Resistance Drift Tolerance on PCM Power and Endurance**

A simple method to improve the readout reliability of MLC-PCM is to adopt an enlarged resistance margin to guard against drift. To accommodate multiple resistance levels, the gap between the minimum and the maximum resistance states needs to be increased. As the resistance value of the highest resistance state is determined by the active volume size of the GST material that is switched to the amorphous state, increasing the level of the highest resistance requires a programming current of higher amplitude. To quantify the increase in programming current, a heat conduction model[57] is used, which captures the flow of heat in the PCM device during programming operations. The model takes the physical dimensions of the PCM device, the properties of the GST material and the target resistance value as inputs and produces an estimated programming current that can achieve this target resistance level. It is assumed that the read current is 40µA and the estimated SET and RESET programming currents for each given margin are listed in Table 5-2. The total power consumption of the PCM storage is investigated and normalized the power to the case in which the smallest margin (e.g. 0uA) is
used. The results are shown in Figure 5-4(A). As can be seen, the total power consumption increases exponentially with the enlarged margin. This is because 1) the total power of PCM is dominated by the write power and 2) the write current is found to increase exponentially with the enlarged margin since the volume size of active GST material switched between states is proportional to the electrical-pulse generated thermal energy, which shows a power-law relation with the write current (i.e. $q_{total} = t^2 R t$). Note that $fma3d$ is less sensitive to the margin size due to the very few writes. Across all simulated benchmarks, we found that the power consumption of the MLC design with a wider margin (i.e. 2uA) is 2.3X as high as that of the design with a very small margin (i.e. 0uA).

Increasing the margin not only consumes more power but also exposes all PCM cells to a much higher programming current. Such high programming power can physically degrade the endurance of phase change material, leading to a reduced lifetime. The endurance test performed in [90] shows a power law relation between cycle lifetime and programming energy: higher programming energy results in degraded endurance. Assuming an endurance of 1E08, as projected by ITRS 2007 [69], compared to PCM cells with a 0uA margin, the energy increases in write can result in an approximate 100X reduction in the endurance when a larger margin (i.e. 2uA) is required. As estimated in [16], an endurance of 1E08 allows an 8-10 years lifetime for PCM and the estimated lifetime can be significantly reduced to as little as 1-2 months (8-10 years / 100 ≈ 1-2 months) in order to tolerate resistance drift. To summarize, if left unaddressed, resistance drift makes it more challenging to realize the reliability goal of MLC-PCM and the conservative design methods negatively impact its power and endurance.
Helmet: Hardening Resistance Drift with Minimized Power and Endurance Impact

In this Section, cross-layer techniques are proposed that span microarchitecture- and system- levels to cost-effectively enhance MLC-PCM resilience to resistance drift.

Data Inversion and Rotation

As drift shows a strong correlation with the data pattern, this motivates the development of an adaptive data inversion and rotation scheme to convert the original pattern to a drift-tolerant one. The key idea is to store the majority of values in their drift-insensitive formats (e.g. “00” or “11”) rather than drift-sensitive format (e.g. “01” or “10”) by selectively applying bit-inversion, rotation, or a combination of both. Among those, bit inversion changes the states of all bits to the opposite states. For example, given the data “0101” stored in two MLCs in Figure 5-5(A), both cells are programmed to the intermediate state “01”. This high-resistance state “01” is more likely to drift to a wrong state than its inverted state “10”, which is of relatively lower resistance level. Therefore, storing the inverted data “1010”, rather than the original one, can decrease the probability of drift-induced readout errors. Note that data inversion doesn’t change the total number of drift-insensitive or sensitive states before and after the inversion. This is because the bit inversion of “00” and “01” is “11” and “10” respectively, and vice versa. Nevertheless, data inversion may reduce the number of highly drift-sensitive states by converting them to less drift-sensitive states when the former dominates the latter in terms of occurrence frequency. Data inversion performs well in this case, whereas its benefit diminishes as these two states become balanced such as data “0110” and “1001”. In such cases, rotation can be used as an alternative to alter the pattern to make it drift friendly. Note that a rotation of an odd number of bit positions can change the bit-pattern stored in a 2-bit-per cell PCM, while rotating by an even number of bit positions cannot. Taking the “0110” in Figure 5-5(B) as an example, right rotation by one bit position changes the data to “0011”. Consequently, the two multi-level cells are programmed to
the RESET and SET states respectively, which are drift-insensitive. The same applies to another
data pattern “1001”. Moreover, similar to data inversion, rotation is also able to reduce the
number of highly drift-sensitive states. For instance, the rotation of “0101” produces “1010”.
Thus, different than inverting, rotating can not only convert highly drift-sensitive state to less
drift-sensitive state but also change the states from drift-sensitive to drift-insensitive. By
applying both inversion and rotation simultaneously, further benefits can be achieved as is shown
in Figure 5-5(C). Although both data inversion and rotation are able to change the bit-pattern to
improve tolerance to drift, they can also degrade the read reliability when applied blindly. For
instance, data inversion may cause increased vulnerability to drift in the case when the less drift-
sensitive state occurs more frequently than the highly drift-sensitive states. Data rotation may
change the state from drift-insensitive to drift-sensitive, such as from “0011” to “0110”.
Therefore, both bitwise operations need to be applied selectively, depending on the pattern
presented in the original data.

In this work, an adaptive data inversion and rotation scheme is proposed which selectively
applies inversion, rotation or the combination of both on the original data and update the PCM
memory array with the resulting data, referred to as coded data. To achieve this, the MLC-PCM
memory controller is augmented with a Data Manipulation Unit (DMU) shown in Figure 5-6,
which performs the pattern inspection and alteration of the original data. When a page is evicted
from DRAM buffer and to be written into the PCM storage, it is divided into data blocks. Each
data block is of the same size as a write queue entry and is inserted into the write queue in the
PCM memory controller. It is assumed that each write queue entry is of the same size as a cache
line. Subsequently, DMU takes the original data in an entry of the write queue as input and
produces encoded data, which is used to replace the original data in the write queue to be written
to memory. Inside the DMU, by applying inversion, rotation or both on the data segment separately, the distribution of logic levels is tracked in each resulting coded data segment as well as the original data segment and a high weight is attributed to drift-insensitive state and a low weight to drift-sensitive states. Then the coded data segment or the original data that has the highest total weight is chosen as the final coded data. Note that [74] proposed a partial write scheme for optimizing PCM power and endurance and it is assumed that a similar scheme is applied on the PCM storage. The data blocks that are modified in DRAM buffer are tracked and only back dirty data blocks are written back to PCM storage when a page is evicted from the DRAM.

In order to restore the coded data back to the original value upon a memory access from the DRAM or write-back to disk, a table, \textit{Invert\_Rotate\_Table}, is used which tracks the status of each coded data block aligned on cache line boundaries, as is shown in Figure 5-6. There are two bits in each entry of the table: \textit{Invert\_bit} and \textit{Rotate\_bit}. The \textit{Invert\_bit} indicates whether the associated coded data segment has been inverted or not (e.g. “1”: inverted and “0”: non-inverted) and \textit{Rotate\_bit} records whether the rotation is performed on it or not (e.g. “1”: rotated by one bit and “0”: non-rotated). The entry of the \textit{Invert\_Rotate\_Table} is updated by DMU, which determines the bit operations performed on the original data. Note that [75] proposed a simple line shifting mechanism, which performs a rotation on a per-byte basis to even out the write within a PCM block, whereas the rotation in our study is on a per-bit basis with the aim of altering the bit-pattern stored in multi-level cells. By using a rotation granularity of 9 bits (1\text{byte}+1\text{bit}=9\text{bits}), both wear-out leveling and pattern change can be achieved simultaneously. Upon a memory reference from the DRAM, a lookup in \textit{Invert\_Rotate\_Table} is performed and the result is loaded into the register \textit{Invert\_Rotate\_Control}, which invokes the necessary
operations to restore the coded data back to its original format. A similar procedure applies for writing back data from the PCM storage to the disk and then the entry in the Invert_Rotate_Table is reset to zero.

To implement the proposed scheme, a dedicated DRAM-based storage inside memory controller is used for Invert_Rotate_Table. For a MLC-PCM with an effective capacity of 2GB and a cache design with 64-byte cache line size, Invert_Rotate_Table requires an 8MB storage capacity with an area overhead of 0.4% of total memory die area and an additional less than 3% of PCM power consumption. Alternatively, the Invert_Rotate_Table can be implemented using a software approach by extending the page table data structure inside the OS. As each page frame contains 64 (4KB per page ÷ 64B per data block = 64) data blocks, the page table entry can be extended by 16Byte (2bits-per-data-block × 64 data-block-per-page ÷ 8bits-per-byte = 16Byte) to store the Invert_bit and Rotate_bit bits. In addition, the TLB entry is also extended by 16Byte so that Invert_bit and Rotate_bit bits can be loaded into TLB to avoid the long latency of table lookup in the memory. To support the data inversion and rotation operations, we adopt a hardware-assisted design with an estimated less than 1% area overhead of memory die area, a latency of 45ns and a 212mW power consumption. Note that the Invert_Rotate_Table lookup is performed simultaneously with the memory access and its latency (i.e. conservatively estimated to be 25ns) is significantly smaller than that of PCM access (i.e. 48ns detailed in the next section). Thus the Invert_Rotate_Table lookup doesn’t increase the latency of each memory access, whereas we assume a 5ns additional latency for each PCM access due to the delay in restoring the coded data to the original one. This extra latency in bit flipping and rotating can cause performance degradation and its power overhead can further exacerbate PCM high power issue. Nevertheless, it is observed a noticeable power and performance improvement. This is
because of two-reasons: 1) the drift-insensitive states require less power and delay in programming since the SCU sequence is only required for intermediate states; 2) the lower readout error rate incurs less performance penalty in correcting errors.

**Hybrid SLC/MLC**

Although MLC provides more capacity, it is susceptible to resistance drift. On the contrary, drift is not critical for SLC and it can even improve the read reliability by enlarging the resistance margin between RESET and SET states. In this work, a hybrid SLC/MLC PCM design is proposed that allows (1) MLCs to operate in the SLC mode, and (2) the operating mode to switch from the MLC mode to the SLC mode on a per-page basis. In this study, it is assumed that a page frame size is 4KB and storing the data of a page frame requires 32K SLCs or 16K 2-bit-per-cell MLCs. Instead of operating in the MLC mode for all page frames in the PCM storage, the mode of page frames is selectively switched from the MLC mode to the SLC mode to trade the capacity for an improved read reliability. Note that there are both SLC- and MLC-memory regions in the PCM storage and consequently the aggregated capacity varies, depending on the size of each region. To minimize the performance degradation due to the reduced capacity of the PCM storage, the reference characteristics to it is monitored and the mode switching is invoked only when the reference shows a drift-sensitive behavior.

A multi-level PCM cell stores multiple states by exploiting additional states provided through partially amorphizing phase change material. By only programming MLCs to fully amorphous (e.g. “00”) and fully crystalline states (e.g. “11”), a multi-level cell can act as a single-level cell and a binary “0” and “1” is represented by state “00” and “11” respectively. To support the hybrid SLC/MLC design, a slight modification in PCM peripheral circuitry is required. For the write driver functional block, when the target memory region operates in the SLC mode, only the partial or the entire initialization sequence of the MLC programming
mechanism is performed and the subsequent SCU sequence is disabled. This is because the PCM cells can be programmed to either the highest-resistance state or the lowest-resistance states when only the initialization sequence is applied. During the read operations, data stored in PCM cells are detected by sense amplifiers as if all cells operate in the MLC mode. Doing so eliminates the hardware overhead for modifying the sense amplifier circuitry. If the cells operate in the SLC mode, the actual data is extracted by discarding every other bit in the data. Otherwise, the whole data is returned without modification. Note that although bandwidth of accessing SLC-pages is as half as that of accessing MLC-pages, the performance impact is negligible because of twofold: 1) the technique of returning the critical data first is widely adopted in memory design; and 2) it is observed that only a small fraction of total pages in PCM operate in SLC mode. Once a page is converted to SLC mode, inversion/rotation is disabled for memory updates on SLC page frames, since they are completely immune to resistance drift. Nevertheless inversion/rotation is enabled before the conversion takes place, since a period of time is required for collecting the run-time memory reference characteristics before identifying the pages that need to be converted to SLC mode.

To keep tracking the mode of page frames in PCM storage, a single bit, \texttt{SLC\_MLC\_MODE}, is used for each page as an indicator of its operating mode (e.g. \texttt{SLC\_MLC\_MODE}=1 for the SLC mode, \texttt{SLC\_MLC\_MODE}=0 for the MLC mode). The bit is stored in the page table entry and managed by the OS. Upon a memory access request is sent to PCM storage, the bit is forwarded to the memory controller to determine the proper operations required to perform read/write in either the MLC or SLC mode. When a page frame is first allocated by the OS in PCM storage, its memory storage mode is set to the MLC mode by default to maximize the memory capacity. To identify the pages in PCM storage that can benefit from
mode switching, two counters, \textit{Last\_write\_cycle} and \textit{Avg\_read\_to\_write\_latency}, are associated with each page to track the average read-to-write latency of the page. The \textit{Last\_write\_cycle} is updated upon a write reference on a page and it records the timestamp of the most recent write. For each read reference, \textit{Avg\_read\_to\_write\_latency} is updated to record the average number of cycles elapsed between the current timestamp and the timestamp of the most recent write stored in \textit{Last\_write\_cycle}. When the \textit{Avg\_read\_to\_write\_latency} is greater than a pre-defined threshold, the memory reference behavior of the associated page is classified to be drift-sensitive. In this work, a threshold of $1E12$ cycles is chosen, which is found to be most effective by the simulations. A mode switch is initiated by migrating the page to a new physical location in the PCM storage, where two physical memory pages (i.e. 32K MLCs) operating in the SLC mode are created. Page migrations are performed transparently to the program with the aid of the OS, which is responsible for maintaining TLB coherence, copying the page to its target address (this migration is emulated in the simulation by invoking a \texttt{bcopy()} routine), flashing the cache lines belonging to the pages to be migrated. For all \textit{Read\_to\_write\_latency} counters, they are halved periodically by shifting right one bit. As a context switch takes place every 10-200ms in a normal Linux OS, 10ms is used as the interval to make the scheme aware of the altered memory reference behavior due to running different programs caused by a context switch. In addition to page allocation, the page replacement policy is also augmented by using a second-chance algorithm to minimize the capacity loss due to the proposed hybrid design when there aren’t plenty of free memory frames available. More specifically, the default OS page reclaiming algorithm is used to select the top ten pages that are ready to be replaced. The page operating in the SLC mode is chosen to be freed or written back to disk if it is dirty. To deal with the addressing issue in the variable PCM storage capacity, the entire physical memory is addressed
as if it is MLC-only PCM. During the address translation, a virtual address is first translated to a physical address as if the memory is MLC-only. Then the page offset bits of the physical address are left-shifted by one bit, if the $SLC_{-_MLC\_MODE}$ is set, to obtain the correct physical address.

Converting a page frame from the MLC to SLC mode improves reliability, but at the cost of reduced capacity. Nevertheless, the lower error rate of the SLC mode may be capable of mitigating memory latency by reducing the delay caused by correcting errors, thereby offsetting the performance degradation due to smaller capacity. The primary hardware overhead for implementing the hybrid SLC/MLC design is the die area required for two counters associated with each page frame. It is estimated that the area overhead is about 0.9% of total memory die area, assuming DRAM-based storage is used. The access latency and power consumption (per access) are estimated to be 35ns and 120mW respectively. All of these overheads have been taken into account in our evaluations.

**Temperature Aware Page Allocation**

Temperature has been shown to have a significant impact on resistance drift [108]. Upon a page allocation request, a conventional OS virtual memory management scheme allocates the next available page(s) without taking the temperature of the physical memory location into consideration. The key idea of the temperature-aware page allocation scheme is to collect the temperature distribution profile of PCM through a thermal sensor network and favor free pages in lower temperature regions than those in higher temperature regions. Moreover, as pages can operate in either SLC or MLC mode, page allocation is used to map drift sensitive pages (e.g. pages operating in the MLC mode) to regions that have low temperature and place drift tolerant pages (e.g. pages operating in the SLC mode) to regions that exhibit high temperature. Note that it is assumed that a 3D die-stacked on-chip memory design is adopted in this case and the same technique can be applied to an off-chip design.
To obtain the runtime temperature profile of a PCM layer, a thermal sensor design is adopted similar to those used in the Intel Core Duo [112]. Figure 5-7 shows an overview of thermal sensor network architecture. The sensing device is an analog thermal diode and the voltage across the diode exhibits strong temperature dependence. This sensing device occupies little area and allows a large number of thermal diodes to be placed across the die. Although the thermal sensors are scattered across the entire memory die, the sensor data processing is centralized. An analog multiplexer is used to probe all sensing devices and therefore the temperature at different parts of the die can be collected. An A/D circuit is used to convert the analog signal from each sensor into a digital reading, which is assumed to be 7 digits representing a temperature range from 0ºC to 128ºC in our study. To achieve measurement accuracy, each sensor is calibrated at test time. This sensor implementation is capable of providing accuracy levels around 1ºC, which is sufficient for our thermal monitoring. Note that the temperature is not evenly distributed across the entire memory layer. For example, the temperature distribution across the memory regions on top of core areas can vary significantly, while those regions atop caches is likely to show a small variation. To capture the spatial variation in temperature and minimize the number of sensing devices required, a non-uniform sensor placement approach is adopted. After dividing the floor plan of the PCM layer uniformly into grids shown in Figure 5-7, a thermal diode is placed in the center of each grid cell that is on top of core area, while a thermal diode is placed in the center of a memory region covering 4 grid cells, if the region is physically located on top of L2 cache. To determine the size of grid cell, thermal analysis is performed across all simulated workloads and adopted a grid size of 1.5mm by 1.5mm. This grid size is able to ensure that the temperature error within any given cell is smaller than 5ºC, since the variation in the resistance drift is small when the temperature varies
less than 5°C [108]. Due to the small size of the sensing device, a total of 90 on-chip thermal sensors are employed on the PCM layer. As temperature doesn’t change significantly over time, it is assumed that a sampling interval of 1s is adopted.

After collecting the run-time temperature, the data is feed back to the OS through an interrupt mechanism and they are consulted by the OS memory management module during page allocation, as is shown in Figure 5-7. Due to its low frequency (e.g. 1Hz), the performance degradation caused by the periodic interrupt is negligible. A page allocation request is generated when a page fault exception occurs since the OS delays allocating dynamic memory to user processes until the page is referenced. A typical technique used in Linux is the buddy memory allocation technique which groups free pages into multiple lists of memory blocks and the memory blocks in each list contain a variable number of contiguous page frames. To satisfy a memory allocation request, a list that contains large enough memory blocks is traversed and a random memory block in this list is selected and returned for allocation. To augment the page allocation scheme with the capability of temperature awareness, the temperature of the memory regions that each memory block belongs to is obtained by looking up the collected runtime temperature. After traversing the list, the memory block that has the lowest temperature is returned to satisfy the memory allocation request. Beside page faults, page allocation requests are also triggered due to page migrations. As described in the previous section, the proposed hybrid SLC/MLC design may initiate a mode switch by migrating a page operating in the MLC mode to a new location where memory cells will operate in the SLC mode. As SLC is insensitive to resistance drift, free pages in the high temperature regions are favored over those in low temperature regions in this case. Doing so provides more opportunity for pages in MLC to be allocated in low temperature regions.
**Experimental Methodology**

In this section, the experimental methodology for evaluating the benefits of the proposed techniques is described. A quad core system with a shared 2MB cache and a DRAM/PCM hybrid memory is simulated. Each core is an out-of-order processor with parameters listed in Table 5-1. For the hybrid memory, the small DRAM buffer is on the processor layer, which is managed by the memory controller and is transparent to the OS, while the MLC-PCM storage is stacked on-top of the processor layer and managed by the OS. For the MLC-PCM, 2GHz frequency and 90nm technology with a supply voltage of 1.6V are assumed. Table 5-2 summarizes the timing and power characteristics of the modeled MLC-PCM. Five MLC designs with different resistance distributions (i.e. Margin0, Margin0.5, Margin 1.0, Margin 1.5 and Margin 2.0) are evaluated, which provide a readout current margin of 0uA, 0.5uA, 1.0uA, 1.5uA and 2.0uA respectively. For each margin size, the minimum, mean, maximum resistance values for each logic state along with its RESET and SET current are shown in Table 5-2. These values are obtained using a one-dimensional heat conduction model[57] that calculates the minimum programming current required for a successful write operation. To evaluate the proposed techniques, applications from SPEC 2000 and NAS parallel benchmark suites are used for simulations. These benchmarks are listed in Table 5-3 along with the drift-induced error rate when they run standalone. 20 programs from SPEC 2000 are selected with reference inputs and 7 programs from NAS Parallel Benchmarks Version 3.2 with Class “C” input data set are used. All benchmarks are compiled on an x86 platform using GCC or FORTRAN compiler with optimization level –O3. To form 4-threaded multiprogramming workloads, all benchmarks are first categorized into high-error-rate (error rate>3%), moderate-error-rate (3%<error rate<1%) and low-error-rate (error rate<1%) groups. In Table 5-4, the High-, Moderate- and Low- error-rate workloads (H1-H3, M1-M3, L1-L3) consist of four benchmarks exclusively from each
category. For all experiments, the SimPoint toolset is used to choose representative execution intervals and perform detailed cycle-level simulation until at least 20 billion instructions are committed on every core. By taking the context switch into consideration, it is assumed that each workload executes for 50ms and other programs are swapped in for execution in the next four consecutive 50ms intervals, since it is observed that context switch occurs every 50ms in a typical operating system. Thus it is found that the total real time for executing each workload has an average of 400 seconds, which is a sufficient long period of time for investigating resistance drift.

For performance and power evaluation, a framework is developed based on a heavily extended full-system simulator, PTLSim/X [98], integrated with a modified DRAMSim [60] memory model, a resistance drift model and the Wattch Power Model [67]. PTLSim/X is a cycle accurate simulator supporting the x86 instruction set architecture. PTLSim/X is extended to model a 2-level write back cache, contention for memory buses and bus traffic. To model the latency and energy of PCM, the DRAMSim timing module is enhanced to model PCM-specific peripheral structures and extended its power module for PCM energy estimation. For the timing and power parameters, the range of values for a given PCM parameter is identified through an extensive literature search and used the median value for that parameter (listed in Table 5-2). To capture the behavior of resistance change, the drift model is used, which estimates the post-drift resistance values based on memory reference patterns and run-time on-chip temperature, to calculate the readout error rate. It is assumed that an advanced error correction code (ECC), Bose-Chaudhuri-Hocquenghem (BCH) code [113], is employed in our MLC-PCM to investigate the impact of readout error rate on performance. This BCH code is assumed to be able to correct an arbitrary number of errors with an extra 2ns latency to correct each additional error. To obtain
the runtime temperature profile, the temperature modeling tool Hotspot 4.0 [70] is used to
dynamically calculate the temperature distribution across the PCM layer. Hotspot takes the
floorplan and the power consumption for each functional block as input and generates accurate
temperature estimation for each block. The floorplan of the core layer and the PCM layer is
similar to the one shown in Figure 5-7. The power trace input to Hotspot is generated using
Wattch and DRAMSim power models. To implement the OS page migration and allocation
scheme, the custom memory manager in PTLSim is modified to support these schemes. This
modified memory manager is responsible for maintaining TLB coherence, copying the page to
its new location (this migration is emulated by invoking a bcopy() routine) and flushing the
cache lines belonging to the pages to be migrated. The associated performance and power
overhead have been taken into account in our reported simulation results.

Results

In this section, the readout reliability, power and endurance benefits of the proposed drift-
tolerant techniques are presented. The experimental evaluation includes: (1) the efficiency of
drift tolerance; (2) the power savings and lifespan extension by adopting drift-tolerant
techniques; and (3) the sensitivity analysis.

Efficiency of Drifting Tolerance

Figure 5-8 shows the readout reliability benefit of the proposed drift-tolerant techniques.
“InvRot” represents the MLC-PCM implementation that utilizes the data inversion and rotation
to tolerate drifting. “InvRot+Hyb” is the design that combines the data inversion and rotation
with the hybrid SLC/MLC design. The last bar, “InvRot+Hyb+PageAlloc” represents the case in
which all of the proposed techniques are applied simultaneously. As shown in Figure 5-8, the
average of the absolute read error rate decreases significantly from 5.1% to less than 0.05% for
the “None” case when the margin is enlarged from 0uA to 2.0uA. Nonetheless, such wide margin
designs improve the readout reliability at the cost of 2.3X power overhead and 100X endurance degradation. For the design with a 0uA margin size, the “None” design incurs an absolute error rate of 9.6% on average across High-error-rate workloads (i.e. H1, H2, H3), while this absolute error rate can be reduced by 3%, or relatively 30%, through data inversion and rotation. This error rate reduction is the result of the reduced number of drift-sensitive states (i.e. “01” and “10”) stored in memory cells. Figure 5-9 shows the percentage of cells that stores “01” or “10” during read references before and after applying data inversion and rotation. It is observed that the number of “01” and “10” states is reduced by a relative 32% and 3% respectively after applying inversion and rotation and the total number of drift-sensitive states is decreased by a relative 20%. Note that the reduction in “01” is significantly higher than that of the “10” state. This is because data inversion and rotation may convert the highly-drift-sensitive state “01” to the less-drift-sensitive state “10”. Across all simulated workloads, H3 has the largest reduction, a relative 62.6%, in the total number of drift-sensitive states and this leads to an absolute 5.7%, or relative 61%, decrease in the error rate for 0uA margin size in Figure 5-8. When the SLC/MLC hybrid design is employed, the “InvRot+Hyb” scheme further reduces the error rate by absolute 1.4%, or relative 48.5%, with respective to the “InvRot” case. This is because the pages that exhibit long read-to-write memory reference characteristics are switched from the MLC mode to the SLC mode, thereby eliminating further readout errors after mode switching. Figure 5-10 shows the percentage of pages that operate in SLC mode. An average of 4.1% of total pages performed a mode switch and this leads to a small 2.1% memory capacity overhead caused by the hybrid design. By combining all proposed techniques together, “InvRot+Hyb+PageAlloc” reduces the absolute error rate from 4.2% to 1.2%, a relative decrease of 69.3%. Across all of the
investigated margins, the “InvRot+Hyb+PageAlloc” achieves an average of 87% error rate reduction over the “None” case.

Compared to the conservative approach, the proposed techniques allow a smaller margin to meet the reliability requirement with lower cost, as is shown in Figure 5-8. For example, to ensure a readout error rate less than 1%, a 1.5uA margin size is required by the “None” design, since the margin has to be tailored to the worst case so that the error rate requirement can be satisfied across all workloads. With the proposed drift-tolerance techniques employed, this margin requirement can be relaxed to 0.5uA, because all workloads show a less than 1% error rate in the “InvRot+Hyb+PageAlloc” case with 0.5uA margin size.

Figure 5-11 shows the normalized harmonic IPC values for the proposed techniques over that of “None” case. The bar labeled “gmean” denotes the geometric mean. Across all simulated workloads, it is observed an up to 2.2% performance improvement for the “InvRot+Hyb+Page” design over the baseline “None” for the 0uA margin size. This is because the small margin size leads to a high error rate and consequently a long latency for correcting errors is required, thereby incurring a performance penalty in the baseline case. The reduced error rate due to our techniques can provide performance benefits that may completely offset the overhead caused by using our techniques. This is especially the case for workloads H3 and M2, which achieve more than 80% reduction in error rate after applying drift-tolerance techniques. On the contrary, a wider margin (i.e. 2uA) limits the benefit that is provided by our techniques and therefore this limited benefit cannot offset the performance overhead associated with the proposed schemes.

**Impact on Power and Endurance**

Figure 5-12 shows the power consumption of the memory system for each margin setting and Figure 5-13 shows the estimated lifespan. All values reported are normalized to the “None”
case with a 0uA margin setting. For power estimation, it is assumed that the write power increases with the enlarged margin while the read power per access remains constant. To estimate lifespan, each workload is executed repeatedly and the number of writes to each bit is tracked. 1×10^5 instead of 1×10^8 writes is used in the lifetime estimation due to the extremely long simulation time to collect 1×10^8 writes to any PCM cell and it is assumed that ECC is disabled. With this accelerated estimation method, when a bit has more than 1×10^5 writes to it, it is marked as a failed cell. A memory access failure is defined as when there is a failed bit in a memory reference. The lifetime of the PCM system is estimated as the number of cycles elapsed before the first memory access failure occurs.

As shown in Figure 5-12, the PCM memory power consumption increases significantly with the increased margin size due to the elevated programming power used to ensure a wider margin. This high write power also causes the exponentially degraded endurance shown in Figure 5-13. As discussed the previous section, to ensure a readout error rate of less than 1% across all simulated workloads, the “None” case requires a margin size of 1.5uA, while the “InvRot+Hyb+PageAlloc” design can achieve the same goal with a smaller margin of 0.5uA. The normalized power and endurance results are summarized for both designs in Table 5-5. As can be seen, the proposed schemes with a small margin size consume 28% (i.e. 1-1.26/1.76 ≈ 28%) less power than that of the “None” case with wider margin. The power increase in the “None” case is a result of the significantly increased power used for each programming operation and the total power consumption of the PCM system is typically dominated by the write power. The elevated programming power per write not only causes higher power consumption but also results in a 15X (i.e. 0.684/0.046 ≈ 15) endurance degradation relative to the “InvRot+Hyb+PageAlloc” case. For a given margin setting, “InvRot” consumes 2%-5% more
power than that of the “None” case due to the power overhead of restoring coded data back to the original value during memory read references, whereas the data inversion and rotation has negligible impact on lifespan. Although “InvRot+Hyb” involves page migration, we found that “InvRot+Hyb” only incurs 4% more power. This is because (1) a small percentage of the pages are switched to MLC mode, thereby invoking a small number of page migrations, and (2) the writes to SLC pages consume less power than that of MLC pages since the stair-case up sequence is not performed. This power savings offset the power overhead associated with page migration. “InvRot+Hyb” shows negligible impact (less than 3%) on endurance degradation of PCM lifespan, since the page migration incurs only one-time endurance overhead. Similar to the “InvRot+Hyb” case, “InvRot+Hyb+PageAlloc” suffers 4% more power overhead than the “InvRot” case due to the additional power consumed by the thermal network and executing OS routines for temperature-aware page allocation. However, “InvRot+Hyb+PageAlloc” doesn’t introduce additional endurance overhead on top of the overhead incurred in the “InvRot+Hyb” case.

**Sensitivity Analysis**

Although bit inversion and rotation can change the data pattern stored in MLCs, the effectiveness of this scheme is sensitive to the granularity on which inversion and rotation are performed. A small granularity allows fine control over the data pattern, but at the cost of significant implementation and performance overhead. A sensitivity analysis is performed and Figure 5-14 shows the normalized error rate as the granularity on which inversion and rotation are performed is decreased from 64Byte, a cache line size used in this study, to 1Byte. Note that the results are normalized to the error rates of the 64-Byte granularity. As can be seen, the geometric mean of the error rate decreases by less than 15% when the granularity reduces to
16Byte. Although the 16-Byte granularity achieves only 10%-30% error rate reduction relative to that of 64-Byte granularity, we estimate a 40% latency increase in each memory read reference and 4X storage overhead. Thus the 64-Byte granularity is selected that provides a good trade-off between reliability benefit and complexity/performance overhead.

**Related Work**

Due to its superior scalability, high density and low power features, phase change memory is attracting increasing attention as one of the most promising technologies for next generation memory. Recently, there has been increasing interest in exploring phase change memory in computer architecture and system design [74][75][99][100][110][114][115]. [74] provides a rigorous survey of phase change memory properties, examines PCM buffer organization and proposes partial writes to tolerate the long latency and high energy of writes. [75] explores the high density feature to increase main memory capacity of future systems with minimum cost and power. A hybrid memory system consisting of PCM storage coupled with a small DRAM buffer is proposed. Such a hybrid memory system is able to combine the low latency benefit of DRAM and high capacity benefits of PCM. [115] proposes and evaluates an adaptive bit flip technique to enhance write performance, energy and endurance. Although PCM can increase memory capacity in a cost-effective and power-efficient manner, the limited endurance imposes a constraint on the lifespan of PCM-based memory. [100] investigates a 3D die-stacked PCM memory with architectural- and OS- support. To maximize PCM lifespan, [75] presents a set of techniques to extend the lifetime of PCM memory, such as redundant bit-write removal, row shifting and segment swapping. Furthermore, [110] propose a simple and effective wear-leveling technique that uses only two registers rather than a large storage in conventional wear-leveling schemes to minimize the area and latency overhead. In addition, [114] characterizes the impact of process variation on PCM systems and proposes several techniques to mitigate the deleterious
impact of process variation for PCM in the upcoming nano-scale processing technology era. This study differs from these work in that we focus on multi-level cell PCM and investigate the impact of its drift phenomena on the readout reliability.

Similar to the proposed approach, [75] used row shifting to shift a row by one byte at a time to even out the writes in a row and [115] adopted bit flipping to combat the high programming power of PCM memory. Data rotation and inversion are used in this work to change the data pattern stored in multi-level cells for reliability purposes. Note that the data rotation adopted in our proposed technique is performed at bit-granularity since the data pattern remains unchanged if data is rotated by one byte. In addition, [116] proposed morphable memory system that dynamically regulates the number of bits per cell in memory system depending on the workload requirement for performance enhancement, while the proposed schemes in this work focuses on the reliability. Like PCM, Flash is also able to provide MLC capabilities, and [63] presents a hybrid SLC/MLC flash disk cache that can take advantage of the low access latency of SLC and the large capacity of MLC. Differing from [63], the proposed scheme switches a page from MLC to SLC mode based on whether the memory reference to this page is drift-unfriendly or not, while [63] invokes the mode switching according to the page access frequency.
Table 5-1. Baseline machine configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>2GHz</td>
</tr>
<tr>
<td>Width</td>
<td>4-wide fetch/decode/issue/commit</td>
</tr>
<tr>
<td>IQ</td>
<td>64 entries</td>
</tr>
<tr>
<td>ITLB</td>
<td>128 entries, 4-way</td>
</tr>
<tr>
<td>Branch Pred.</td>
<td>2K entries Gshare, 10-bit global history</td>
</tr>
<tr>
<td>BTB</td>
<td>2K entries, 4-way</td>
</tr>
<tr>
<td>RAS</td>
<td>32 entries RAS</td>
</tr>
<tr>
<td>L1 I-Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle</td>
</tr>
<tr>
<td>ROB</td>
<td>128 entries</td>
</tr>
<tr>
<td>LDQ/STQ</td>
<td>48 entries</td>
</tr>
<tr>
<td>Int. ALU</td>
<td>4 I-ALU, 2 I-MUL/DIV, 1 Load/Store</td>
</tr>
<tr>
<td>FP ALU</td>
<td>2 FP-ALU, 2 FP-MUL/DIV/SQRT</td>
</tr>
<tr>
<td>DTLB</td>
<td>256 entries, 4-way</td>
</tr>
<tr>
<td>L1 D-Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 3 cycle</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>Shared 2MB, 8-way, 64 Byte/line, 12 cycle</td>
</tr>
<tr>
<td>Hybrid Memory</td>
<td>DRAM buffer (64MB), MLC-PCM(2GB effective capacity, 8 banks)</td>
</tr>
<tr>
<td>Write Buffer</td>
<td>32 entries, 64B per entry</td>
</tr>
</tbody>
</table>

Table 5-2. MLC PCM parameters

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read</td>
<td>48ns</td>
</tr>
<tr>
<td>Write</td>
<td>4000ns[5]</td>
</tr>
<tr>
<td>Power Parameters</td>
<td></td>
</tr>
<tr>
<td>Read current</td>
<td>40uA</td>
</tr>
<tr>
<td>Read voltage</td>
<td>1.6V</td>
</tr>
<tr>
<td>RESET voltage</td>
<td>1.6V</td>
</tr>
<tr>
<td>SET voltage</td>
<td>1.2V</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Resistance</th>
<th>Margin0 (0uA)</th>
<th>Margin0.5 (0.5uA)</th>
<th>Margin1.0 (1.0uA)</th>
<th>Margin1.5 (1.5uA)</th>
<th>Margin2.0 (2.0uA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>“11” Min</td>
<td>4K</td>
<td>4K</td>
<td>4K</td>
<td>4K</td>
<td>4K</td>
</tr>
<tr>
<td>Mean</td>
<td>8K</td>
<td>8K</td>
<td>8K</td>
<td>8K</td>
<td>8K</td>
</tr>
<tr>
<td>Max</td>
<td>13K</td>
<td>13K</td>
<td>13K</td>
<td>13K</td>
<td>13K</td>
</tr>
<tr>
<td>“10” Min</td>
<td>33K</td>
<td>34K</td>
<td>34K</td>
<td>34K</td>
<td>34K</td>
</tr>
<tr>
<td>Mean</td>
<td>73K</td>
<td>73K</td>
<td>74K</td>
<td>74K</td>
<td>74K</td>
</tr>
<tr>
<td>Max</td>
<td>112K</td>
<td>112K</td>
<td>113K</td>
<td>114K</td>
<td>114K</td>
</tr>
<tr>
<td>“01” Min</td>
<td>132K</td>
<td>137K</td>
<td>141K</td>
<td>146K</td>
<td>148K</td>
</tr>
<tr>
<td>Mean</td>
<td>287K</td>
<td>298K</td>
<td>306K</td>
<td>317K</td>
<td>321K</td>
</tr>
<tr>
<td>Max</td>
<td>442K</td>
<td>460K</td>
<td>471K</td>
<td>488K</td>
<td>499K</td>
</tr>
<tr>
<td>“00” Min</td>
<td>463K</td>
<td>545K</td>
<td>612K</td>
<td>731K</td>
<td>782K</td>
</tr>
<tr>
<td>Mean</td>
<td>1006K</td>
<td>1185K</td>
<td>1330K</td>
<td>1589K</td>
<td>1700K</td>
</tr>
<tr>
<td>Max</td>
<td>1549K</td>
<td>1824K</td>
<td>2048K</td>
<td>2447K</td>
<td>2618K</td>
</tr>
<tr>
<td>RESET current</td>
<td>197uA</td>
<td>212uA</td>
<td>232uA</td>
<td>263uA</td>
<td>300uA</td>
</tr>
<tr>
<td>SET current</td>
<td>138uA</td>
<td>148uA</td>
<td>163uA</td>
<td>185uA</td>
<td>210uA</td>
</tr>
</tbody>
</table>
Table 5-3. Benchmarks used to form workloads

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>gzip</td>
<td>20.17%</td>
</tr>
<tr>
<td>bzip</td>
<td>11.21%</td>
</tr>
<tr>
<td>art</td>
<td>8.64%</td>
</tr>
<tr>
<td>swim</td>
<td>7.96%</td>
</tr>
<tr>
<td>equake</td>
<td>7.88%</td>
</tr>
<tr>
<td>ammp</td>
<td>3.85%</td>
</tr>
<tr>
<td>lucas</td>
<td>3.14%</td>
</tr>
<tr>
<td>sixtrack</td>
<td>3.08%</td>
</tr>
<tr>
<td>vortex</td>
<td>3.05%</td>
</tr>
<tr>
<td>UA</td>
<td>2.97%</td>
</tr>
<tr>
<td>mcf</td>
<td>1.78%</td>
</tr>
<tr>
<td>gap</td>
<td>1.91%</td>
</tr>
<tr>
<td>applu</td>
<td>1.61%</td>
</tr>
<tr>
<td>mgrid</td>
<td>1.53%</td>
</tr>
<tr>
<td>facerece</td>
<td>1.42%</td>
</tr>
<tr>
<td>Crafty</td>
<td>1.27%</td>
</tr>
<tr>
<td>BT</td>
<td>1.07%</td>
</tr>
<tr>
<td>perlbmk</td>
<td>1.06%</td>
</tr>
<tr>
<td>EP</td>
<td>0.95%</td>
</tr>
<tr>
<td>mesa</td>
<td>0.92%</td>
</tr>
<tr>
<td>FT</td>
<td>0.91%</td>
</tr>
<tr>
<td>MG</td>
<td>0.03%</td>
</tr>
<tr>
<td>vpr</td>
<td>0.02%</td>
</tr>
<tr>
<td>CG</td>
<td>0.02%</td>
</tr>
<tr>
<td>LU</td>
<td>0.02%</td>
</tr>
<tr>
<td>fma3d</td>
<td>0.01%</td>
</tr>
<tr>
<td>galgel</td>
<td>0.01%</td>
</tr>
</tbody>
</table>

Table 5-4. Workloads

<table>
<thead>
<tr>
<th>Type</th>
<th>Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>H1(gzip, swim, lucas, sixtrack)</td>
</tr>
<tr>
<td></td>
<td>H2(bzip, equake, sixtrack, ammp)</td>
</tr>
<tr>
<td></td>
<td>H3(art, ammp, vortex, gzip)</td>
</tr>
<tr>
<td>Moderate</td>
<td>M1(UA, applu, crafty, BT)</td>
</tr>
<tr>
<td></td>
<td>M2(mcf, mgrid, BT, facerece)</td>
</tr>
<tr>
<td></td>
<td>M3(gap, facerece, perlbmk, UA)</td>
</tr>
<tr>
<td>Low</td>
<td>L1(EP, MG, LU, fma3d)</td>
</tr>
<tr>
<td></td>
<td>L2(mesa, vpr, fma3d, CG)</td>
</tr>
<tr>
<td></td>
<td>L3(FT, CG, galgel, EP)</td>
</tr>
</tbody>
</table>
Table 5-5. A comparison of normalized power and endurance between two schemes that achieves the same error rate

<table>
<thead>
<tr>
<th></th>
<th>Power</th>
<th>InvRot+Hyb+PageAlloc(0.5uA Margin) (Power reduction over “None”)</th>
<th>Endurance</th>
<th>InvRot+Hyb+PageAlloc(0.5uA Margin) (Endurance improvement over “None”)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None (1.5uA margin)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H1</td>
<td>1.76</td>
<td>1.29 (27%)</td>
<td>0.045</td>
<td>0.67 (15X)</td>
</tr>
<tr>
<td>H2</td>
<td>1.77</td>
<td>1.23 (31%)</td>
<td>0.044</td>
<td>0.680 (15X)</td>
</tr>
<tr>
<td>H3</td>
<td>1.66</td>
<td>1.53 (8%)</td>
<td>0.063</td>
<td>0.751 (9X)</td>
</tr>
<tr>
<td>M1</td>
<td>1.77</td>
<td>1.24 (30%)</td>
<td>0.044</td>
<td>0.679 (12X)</td>
</tr>
<tr>
<td>M2</td>
<td>1.77</td>
<td>1.23 (31%)</td>
<td>0.043</td>
<td>0.678 (15X)</td>
</tr>
<tr>
<td>M3</td>
<td>1.78</td>
<td>1.20 (33%)</td>
<td>0.043</td>
<td>0.679 (15X)</td>
</tr>
<tr>
<td>L1</td>
<td>1.78</td>
<td>1.18 (34%)</td>
<td>0.042</td>
<td>0.678 (16X)</td>
</tr>
<tr>
<td>L2</td>
<td>1.75</td>
<td>1.28 (27%)</td>
<td>0.042</td>
<td>0.670 (16X)</td>
</tr>
<tr>
<td>L3</td>
<td>1.76</td>
<td>1.18 (33%)</td>
<td>0.047</td>
<td>0.685 (15X)</td>
</tr>
<tr>
<td>Geomean</td>
<td>1.76</td>
<td>1.26 (28%)</td>
<td>0.046</td>
<td>0.684 (15X)</td>
</tr>
</tbody>
</table>
Figure 5-1. An illustration of the MLC programming mechanism[8].

Figure 5-2. (a) Resistance drift of a PCM device over time (b) Readout reliability issue due to resistance drift (c) Addressing read reliability via increasing margin.

Figure 5-3. The dependence of read reliability on resistance margin between two adjacent states for the low-temperature off-chip design and high-temperature 3D on-chip design.
Figure 5-4. The dependence of (a) power consumption and (b) endurance on resistance margin.

Figure 5-5. Using bit-inversion, rotation or the combination of both to convert the original data to drift-tolerant one.

Figure 5-6. An overview of microarchitectural support for data inversion and rotation.
Figure 5-7. An overview of the thermal sensor network and temperature-aware page allocation.

Figure 5-8. A comparison of absolute readout error rate.

Figure 5-9. The breakdown percentage of state “01” and “10” before and after applying inversion and rotation.

Figure 5-10. The percentage of pages that operate in the SLC mode when the 0uA margin is used.
Figure 5-11. The impact of reduced error rate on performance

Figure 5-12. A comparison of power consumption.

Figure 5-13 A comparison of endurance.

Figure 5-14. Sensitive analysis of inversion/rotation granularity.
CHAPTER 6
ENHANCING THE VERSATILITY, FUNCTIONALITY AND EFFICIENCY OF MULTI-CORE CACHE ARCHITECTURE USING NON-VOLATILE MEMORY BASED CELL DESIGN

The use of caches in microprocessor design has been a widely adopted method to address long memory access latency. Typically, caches are implemented using SRAM due to its fast access speed. To represent the binary values, SRAM stores charges. When information stored in the SRAM cells is destroyed, data has to be retrieved from the low-level in memory hierarchy, resulting in significant performance penalty and power overhead. Moreover, as CMOS processing technologies scale down, the leakage-induced stand-by power of SRAM becomes an increasing concern. In the past, numerous techniques have been proposed to optimize the efficiency of SRAM-based caches. These techniques, however, often target on optimizing a specific aspect of the cache design. Since performance, power and reliability are all converging as important design challenges; it becomes more crucial to explore cache architectures that can flexibly support design optimizations in more than one domain. Moreover, in light of multi-/many- core computing and virtualization, the processor concurrently executes a large number of threads/applications with drastically different characteristics. This trend demands a versatile and flexible cache substrate that allows multiple caching functionalities and optimizations to be simultaneously triggered at run-time. The conventional cache architectures are incapable of achieving the above goals. The objective of this work is to develop versatile cache architectures that can simultaneously enhance the performance, power and reliability of caches while allowing the co-existing of multiple and versatile caching functions and optimizations.

Emerging memory technologies open new opportunities for cache optimizations. For example, [77][117] propose hybrid cache architectures consisting of SRAM and non-volatile memories to improve the performance and power efficiency of caches. In their hybrid cache
design, the non-volatile memory cells and the SRAM cells are used separately and in isolation. In other words, levels of cache hierarchies are built from disparate memory technologies, but cells in a single hierarchical level or bank are designed using the same memory technology. This work moves one step further by advocating a novel, Non-Volatile SRAM (NV-SRAM) design that synergically integrates Magnetic Tunnel Junction (MTJ) based non-volatile memory devices into the standard SRAM cells. The proposed NV-SRAM cell design consists of a conventional six-transistor SRAM cell, two MTJs stacked on top of the SRAM and two extra transistors. The MTJs are connected to the SRAM partition through the two extra access transistors and they serve as backup storage for the SRAM cell or additional storage besides the SRAM cell, depending on the operation mode. When MTJs are used as backup storage, This proposed NV-SRAM design allows data stored in the SRAM to be copied into the coupled non-volatile devices by a store operation and the stored value to be restored back to the SRAM by a recall operation. Alternatively, when the MTJs are used as additional storage, the NV-SRAM design allows two different values to be stored into SRAM and MTJs separately. Compared to the conventional SRAM and non-volatile memory cells, the proposed NV-SRAM provides several benefits: (1) Unlike non-volatile memory, the read and write to the NV-SRAM remain high-speed (e.g. 10-50X faster than non-volatile memories), which is inherited from SRAM; (2) Unlike SRAM, the NV-SRAM offers non-volatility, which stems from the characteristics of non-volatile memory devices and allows data to be preserved without consuming leakage power; (3) The NV-SRAM increases effective storage capacity due to the capability of recording different data in non-volatile memory storage and SRAM storage simultaneously; (4) The tight coupling of the SRAM and the non-volatile devices in each cell facilitates the data transfer between them and allows instance backup and recovery of cache data, thereby avoiding long latency of data refilling from
the lower level of memory hierarchies. Consequently, cache architectures built using NV-SRAM not only share the high-speed characteristics of conventional SRAM but also provide versatile and enriched features, including power-saving, non-volatility and instant data recovering, which make it suitable for a wide range of optimizations (as shown by the case studies presented in the following sections).

**Magnetic Tunnel Junction (MTJ) based Non-Volatile Memory**

Magnetic Tunnel Junction (MTJ) is a class of non-volatile memory devices that store information using their magneto-resistive states instead of electrical charges. This device features non-volatility, high read/write speed, nearly unlimited read/write endurance and good compatibility with CMOS process. A MTJ device consists of two magnetic layers separated by a thin dielectric tunnel barrier layer, as shown in Figure 6-1. The tunnel barrier layer is made up of a conducting but non-magnetic material, typically MgO. One of the magnetic layers (pinned layer) has a fixed magnetic moment direction, while the other layer (free layer) can change its direction via an external electromagnetic field or a spin-transfer torque. The resistance of the MTJ is either low or high, depending on the relative direction (i.e. parallel or anti-parallel) of the free layer with respect to that of the pinned layer. When the two magnetic layers have the opposite directions (the anti-parallel state shown in Figure 6-1(B)), the MTJ exhibits high resistance, indicating a state “1”. In contrast, when the two layers have the same direction (the parallel state shown in Figure 6-1(A)), the MTJ features a low resistance, indicating a state “0”. Spin Transfer Torque (STT) can be used to alter the magnetic moment direction of the free layer. STT is a technique that involves passing current through MTJ devices, where the current is polarized and used to alter the magnetic moment direction of the free layer. To perform a write operation on MTJ, either a positive voltage is applied between the top and the bottom electrodes for writing a “0” (Figure 6-1(A)) or a negative voltage is used for writing a “1” (Figure 6-1(B)).
The current amplitude and duration required to achieve a successful state transition are determined by the material of the tunnel barrier layer. In this work, it is assumed that a MTJ device with a threshold current of 120uA in amplitude and 10ns in duration.

Note that this NV-SRAM design is orthogonal to the type of non-volatile memory. In this work, Magnetic Tunnel Junction based non-volatile memory is used since Flash [63], Phase Change RAM (PCM) [78][74][75] and Ferroelectric RAM (FRAM) [119] all have limited write endurance, which prevents their use for high-performance caches in which data needs to be frequently written. In addition, the write latencies of Flash (1us for NOR Flash [69]), PCM (50/120ns [69]) are much higher than that of a MTJ device (10ns [77]) and programming Flash and PCM requires complex peripheral circuits as well. On the contrary, the Spin Transfer Torque scheme used for programming MTJ doesn’t require additional peripheral circuits and considerably reduces the design complexity and cost.

Non-Volatile SRAM (NV-SRAM)

In this section, the circuit design and operation modes of NV-SRAM are described. Then the performance and power characteristics of cache architecture that is built from NV-SRAM cells are investigated using detailed circuit-level modeling and simulations.

NV-SRAM Cell: Organization and Operation Modes

Figure 6-2 shows the schematic design of the NV-SRAM cell, which consists of a six-transistor (6T) SRAM, a pair of MTJs and two NV-storage access transistors. The two MTJ devices are connected to (or disconnected from) the two nodes (Q and QB) of the SRAM via two NV-storage access transistors, which are controlled by STORE/RECALL (SR) signal. When the “SR” is low, the two access transistors are switched to the off state and therefore the two MTJ devices are isolated from the SRAM portion of the cell. When the “SR” is high and the MTJ devices are connected to the two nodes (Q and QB) of the SRAM, data can be stored from
SRAM to MTJ devices or recalled from MTJ devices to SRAM. “CTRL” connects to the bottom-electrodes of MTJ devices and is used to control the direction of currents passing through MTJ devices during a store operation and serves as power source during a recall operation.

Figure 6-3 illustrates three different modes of NV-SRAM operations: NORMAL, STORE, and RECALL. $V_{\text{high}}$ and $V_{\text{low}}$ represent full supply voltage level (i.e. 1.1V) and zero supply voltage level (i.e. 0V) respectively. In the NORMAL mode, the NV-SRAM cell works like a conventional SRAM cell and data in the SRAM is read or written in the same way as they are performed in a standard SRAM cell. In this mode, the “SR” signal is driven low and MTJ devices are disconnected from the SRAM portion of the cell. Note that data held in MTJs cannot be accessed in this mode. In the STORE mode, MTJs are connected to the SRAM, allowing data in the SRAM to be backed up in the MTJs storage. To achieve this, the “CTRL” is held at the half of the full supply voltage ($V_{\text{high}}/2$) and the “SR” is pulled high to connect MTJs to nodes Q and QB. As shown in Figure 6-3(B), assuming the nodes Q and QB store “0” and “1” respectively, transistors M2 and M4 are turned off, while transistors M1 and M3 are in the on-state. In this case, a negative voltage is applied between top and bottom electrodes of the MTJ connected to node Q, while a positive voltage is applied on the other MTJ. As a result, the MTJ on data “0” side (node Q) is programmed to a high resistance (representing state “1”) as the write current flows from “CTRL” to ground through MTJ and transistor M1. The other MTJ on data “1” side (node QB) is programmed to low resistance (representing state “0”) as the write current originated from $V_{\text{DD}}$ passes through transistor M3 and MTJ and ends at “CTRL” line. The paths of both write currents are shown in Figure 6-3(B). Therefore, the logic states of the SRAM cell are check-pointed as high-low resistance of the MTJs by this STORE operation. Note that the data stored in MTJs is the reversed value of that in SRAM and MTJs are disconnected from the
SRAM at the end of the STORE operation. In order to restore the previous logic status to SRAM, a RECALL is performed to reconnect MTJs to nodes Q and QB and restore the state back to the SRAM. As shown in Figure 6-3(C), this can be achieved by (1) lowering the $V_{DD}$ of target cells to near zero using supply voltage gating technique, (2) driving “SR” to high, and (3) raising “CTRL” to $V_{\text{high}}$. Due to the asymmetric resistance of the two MTJ devices, the data node (node QB) connected to the low-resistance MTJ is charged faster than the other (node Q) connected to the high-resistance MTJ. As a result, voltage at node QB rises faster than that at the other node. The $V_{DD}$ is raised to $V_{\text{high}}$ and the latch function of the SRAM amplifies the voltage difference between the two complementary storage nodes and latches the final state. Consequently, logic “0” is restored on node Q and logic “1” is restored on node QB, which is the same logic state saved by the previous STORE operation. Note that the NV-storage access transistors are turned off by driving “SR” to low, thereby disconnecting MTJ devices from the SRAM portion before returning to the NORMAL operation mode. The STORE/RECALL operations can be invoked by control logics implemented as part of the peripheral circuits of NV-SRAM based cache.

**Performance, Power and Area Characterization of NV-SRAM based Cache**

To model the NV-SRAM cell, a HSPICE circuit model is built to obtain its electrical characteristics. A 65nm CMOS process technology with a supply voltage of 1.1V is assumed in this work. The physical MOSFET model used is BSIM 65nm [43]. To model the electrical characteristics of MTJ devices, the circuit model takes two parameters - the write pulse duration and threshold current of MTJ devices - as input. The range of these parameters is identified through an extensive literature search and the median value for each parameter is used. In this work, write pulse duration of 10ns[77] is assumed. The size of MTJs is assumed to be 100nm×150nm [120] with a critical current density of $8 \times 10^5 \text{ A/cm}^2$ [118] and the derived
The threshold current is approximately 120uA, which can be provided by transistors fabricated using existing CMOS technologies [118].

The functionality of NV-SRAM cell is verified and its performance and power characteristics are analyzed. When the NV-SRAM cell operates in the NORMAL mode, the SRAM portion of the cell is not affected by the non-volatile elements and therefore the read/write access latency and active power are the same to those of a conventional SRAM. As the non-volatile storage access transistors cannot be completely turned off in the NORMAL mode, there are two leakage paths: one leakage current flows from the supply voltage to the “CTRL” line through the PMOS and MTJ device on the data “1” side and the other leakage current flows from the “CTRL” line to the ground through the MTJ device and the NMOS on the data “0” side. It is estimated that the additional leakage is approximately 3nA per cell, which increases less than 5% leakage in total. In the case of STORE operations, the store latency is primarily determined by the write duration of MTJs, which is assumed to be 10ns. The energy required for a STORE operation is 7X of that on a SRAM write operation due to the larger current required for programming MTJs. Figure 6-4 shows the HSPICE simulation waveform of restoring data from the non-volatile elements back to the SRAM upon a RECALL operation. As the “SR” is driven to 1.1V and the “CTRL” starts rising to a full supply voltage level, nodes Q and QB are charged by “CTRL” at a different speed. The voltage difference between Q and QB is then amplified and the resulting states are latched by the SRAM when its supply voltage (VDD) is raised to the full supply voltage level. As can be seen from Figure 6-4(D), data in MTJs can be retrieved in less than 0.5ns. After incorporating the delay of the peripheral circuits, it is estimated that the overall latency of a RECALL operation is within 1ns.
Adding two transistors and two MTJ devices will inevitably introduce die area overhead. To minimize this area overhead and achieve the maximum density, the NV-SRAM cell can be implemented by stacking MTJ devices on top of the SRAM to avoid occupying silicon die area. Figure 6-5(A) shows the three-dimensional structure of the NV-SRAM cell. Note that MTJs and SRAM are both on the same die and connected through metal-to-metal vias. Since the 6T-based SRAM (1.54um×1.12um = 1.288um²) has a considerably larger area than the total area (0.1um×0.15um×2 = 0.03 um²) required by the two MTJ devices, the die area of NV-SRAM mainly depends on the total area of the SRAM portion and the two extra access transistors, whose channel width can be set as small as 130nm. To quantify the area overhead, both a conventional SRAM cell (Figure 6-5(B)) and a NV-SRAM cell (Figure 6-5(C)) are laid out using the Cadence design tool. Note that there are various ways to layout a SRAM and the one adopted in this work shown in Figure 6-5(B) can achieve high density since most of the contacts are shared. The memory cell layouts are based on TSMC 0.18um technology, which is the smallest feature size available to the academia. After scaling down to 65nm, the estimated dimension of our NV-SRAM cell is 1.75um×1.12um. Compared to the dimension of a SRAM cell (1.54um×1.12um), the NV-SRAM cell increases the area of each cell by 13.6% due to incorporating the two NV-storage access transistors.

Figure 6-6 illustrates the overall cache architecture built from NV-SRAM cells. In the data array, each SRAM cell is replaced by a NV-SRAM cell. The bottom electrodes of all MTJs within a cache line are connected to the “CTRL” and the gate terminals of all NV-storage access transistors within a line are connected to the “SR” line. Doing so allows a STORE/RECALL operation to transfer the contents of an entire cache line between SRAM storage and MTJ storage in a single operation. To support versatile operation modes, few additional peripheral
circuits are required for each cache line. The “VDD_Controller” signal switches the supply power on/off to the data array using supply voltage gating technique. The “SR_Controller” signal is pulled up when either a STORE or a RECALL operation is required and the exact operation to be performed depends on the value of “CTRL_Controller”. One bit data_stored is associated with each line to indicate whether the non-volatile storage has a copy of data in the SRAM cells. This bit is set when a STORE operation is performed and is reset when the cache line is invalidated or replaced. The gate-level estimation shows that the area overhead of these additional peripheral circuits is about 1.5% of the entire cache area.

To model the latency and power of a NV-SRAM cache, the CACTI 6.0 [121] is modified and the HSPICE simulation results are used as additional input parameters. Table 6-1 shows a comparison between a 4MB SRAM cache and a 4MB NV-SRAM based cache in terms of area, access latency, dynamic energy and leakage power consumption. Note that the area overhead of the NV-SRAM based cache is about 10%, which is less than that of the NV-SRAM cell since the NV-SRAM cells are only applied to the data array. When the NV-SRAM cache operates in the NORMAL mode, its read/write latency and energy are almost the same as those of a conventional cache. The leakage of NV-SRAM cache increases by 4.3% due to the additional leakage paths introduced by the access transistors, MTJs and “CTRL” line. The latency and the energy consumption of a STORE operation are 10.7ns and 6.2nJ respectively. The high energy consumption is mainly caused by the high write current of MTJ devices. Compared to the STORE operation, the latency of a RECALL operation is much smaller and the dissipated energy is significantly lower. This is because the MTJ devices are not programmed during RECALL operations and bit and bit_bar lines are not charged or discharged.
Exploring the Benefits of NV-SRAM based Caches

The proposed NV-SRAM cell combines the fast speed of a standard SRAM with the non-volatile nature of MTJ devices and offers a variety of features, including low stand-by power, large cache capacity, instance data backup and recovery. In this section, these benefits are explored in cache efficiency optimizations.

Reducing Context Switch Overhead due to Cache Interference

Context switching is a mechanism that is used to share a processor across multiple processes. However, it introduces performance overhead. In addition to the cost of saving and restoring processor registers and executing the OS scheduler code, the cache interference between multiple processes sharing the same cache also contributes to the cost of context switch. The extra cache misses due to cache interference is referred as cache interference misses in this work. Prior work suggests that the cost of cache interference misses is much greater than all other costs [122][123][124]. As the cache interference misses can degrade performance substantially, this work proposes to use NV-SRAM for mitigating the performance overhead of context switching. The basic idea is to utilize the non-volatile elements in NV-SRAM cells as additional storage capacity to perform instant backing up and restoring cache lines when context switching occurs.

Figure 6-7 shows a case in which two processes share a single processor. As the capacity of non-volatile storage is limited, non-volatile storage can be used to accommodate cache line data for only one process at a time. Assuming that process A is the one that needs to be optimized for performance, this process then has exclusive access to the non-volatile storage. Upon switching out process A, a STORE operation is initiated to backup its cache line data from SRAM to the non-volatile storage. To identify cache lines belonging to process A, one tag bit, $Acce_{NVStorage}$, is added to each cache line. The bit is set when a cache line is fetched from
memory by process A or when a cache line is refilled by a RECALL operation. All 
*Acce_NVStorage* bits are reset whenever process A is switched out by the OS scheduler. When 
process A resumes execution, its cache line data in the non-volatile storage can be restored back 
to SRAM by a RECALL operation, thereby avoiding reloading its working set from lower levels 
of memory hierarchy and reducing cache misses. A side effect of RECALL operation is that data 
residing in SRAM are destroyed by the RECALL operation. To avoid over-writing dirty data 
which haven’t been written back to memory, RECALL operations are performed only on clean 
cache lines, not dirty ones. Note that STORE and RECALL operations introduce minimum 
performance overhead by themselves, since data is copied between SRAM and non-volatile 
elements in parallel.

In a case that multiple processes share a single processor but no specific process requires 
for context switch optimization, a QoS-aware policy is proposed that can dynamically choose the 
process that most likely benefits from this optimization and allows it to take the advantage of 
non-volatile storage. To achieve this goal, a metric, *average reuse probability* (*P_{avg\_reuse}*), is used 
as an indicator of the number of context interference misses that can be reduced. One may expect 
that the working set size is the only factor that correlates with context switch cache misses. The 
simulation results (detailed in later sections) show that a process’s working set evicted by other 
processes may not be reused again after it resumes execution. Hence, a good metric should take 
both temporal reuse pattern and working set size into consideration. The temporal reuse pattern 
of an application can be captured by stack distance profiling technique through static analysis 
[125][126] or runtime monitoring [127]. A hardware-implemented, low-overhead, on-line 
monitoring scheme proposed in [127] is adopted in this work. For a N-way M-set associative 
cache, this scheme records the number of hits to N positions in the LRU stack by using N
counters \((C_{1,i}, C_{2,i}, \ldots, C_{N,i})\) in the order of MRU to LRU) for the \(i\)th set and we add another counter \((C_{N+1,i})\) to track the number of misses in this set \((i=1, \ldots, M)\). Hence, \(P_{\text{avg\_reuse}}\) can be computed as follows:

\[
P_{\text{avg\_reuse}} = \sum_{j=1}^{M} \sum_{i=1}^{N} C_{i,j} \times i + C_{N+1,j} \times N
\]

(Eq 6-1)

where \(M\) is the number of cache sets and \(N\) is the number of associativity ways. The OS process scheduler can be extended to associate a \(P_{\text{avg\_reuse}}\) with each process and the \(P_{\text{avg\_reuse}}\) is updated whenever a user process is switched out followed by resetting all counters to zero. Note that the working set size is reflected in \(P_{\text{avg\_reuse}}\) by properly weighting the value of each counter (i.e. \(C_{i,j} \times i\)), since \(C_{i,j}\) equals zero if the \(i\)th cache line in the \(j\)th set has never been accessed. The process with the highest \(P_{\text{avg\_reuse}}\) is identified as the one that most likely benefits from the reduced cache interference misses. The die area overhead of this on-line monitoring scheme is estimated to be 6.2%, assuming each counter is 4Bytes and a cache line size of 64Bytes.

Although the cost of cache interference misses is significant, it is possible that a process can amortize this cost over a long run. The context switching frequency is measured on a real machine running an unmodified Linux and two SPEC benchmarks (\texttt{bzip} and \texttt{ammp}). The measurement shows that a context switch occurs every 15M cycles on average. Assuming a process with an IPC of 1 running on a 2.8GHz processor with a 4MB L2 cache and the memory access penalty is 500 cycles, the maximal possible speed up for the process would be approximately 3.27X if cache interference misses were completely eliminated. Recently, [117] proposed a hybrid SRAM/MRAM cache design, which increases cache capacity by leveraging the high-density MRAM to reduce cache misses. Compared to the hybrid cache design, the proposed NV-SRAM based cache allows cache lines to be placed in a separate storage attached
to the same cache level rather than forcing them to be placed in a lower cache level. This not only reduces the capacity pressure on the lower level cache, but also makes subsequent accesses faster since typically restoring data has lower latencies than accessing a larger non-volatile cache at the lower level. Furthermore, cache lines are recalled simultaneously rather than being fetched sequentially.

**Reducing Cache Leakage Power**

As feature size scales, leakage power starts to dominate the total power consumption of on-chip caches. [128] proposed a drowsy cache scheme to reduce the leakage power by putting cache lines into a state preserving, low-power drowsy model through dynamic voltage scaling. As the non-volatile elements in NV-SRAM cells can retain information without power supply, this work proposes the use of NV-SRAM to achieve leakage power reduction by backing up data into the non-volatile elements and completely powering off cache lines. In contrast to this proposed design, drowsy cache achieves low stand-by power at the cost of increasing soft error vulnerability, since the soft error rate is inversely proportional to the supply voltage. On the contrary, this proposed cache design is able to overcome the vulnerability issue faced by the drowsy cache yet achieve desirable leakage power reduction. This is because that the non-volatile MTJ device is not charge based and hence it is completely immune to soft error strikes.

The additional circuits required implementing non-volatile, low standby power cache is similar to that of a drowsy cache. One bit, $power_{mode}$, is added for each cache line to control the supply voltage to the NV-SRAM cells within the cache line. Thus, the supply voltage to each cache line can be individually switched between the full supply voltage and zero, depending on the state of the associated $power_{mode}$ bit. When a cache line is required to be turned off, no STORE operation is needed if non-volatile storage contains a copy of cache line data (i.e. $data_{stored}=1$ in Figure 6-6) and the cache line is clean. Otherwise, a STORE operation is
invoked to copy the data from SRAM cells to non-volatile elements and sets the *data_stored* bit to “1”. Subsequently, the *power_mode* bit is set to “0” and the supply voltage to this cache line is turned off. When a cache line is accessed in the power-down mode, the *power_mode* bit is set to “1” and the supply voltage to this cache line is switched to full voltage level. Meanwhile, a RECALL operation is performed to restore the previous logic states to the SRAM cells in parallel with the recovery of the supply voltage (as shown in Figure 6-4). As both voltage transitions for each cache line and the RECALL operation can be completed in one cycle with small power overhead similar to that of drowsy cache, the penalty of wake-up is very small. To manage the power mode of all cache lines, we adopt a simple policy which periodically turns off the entire NV-SRAM cache. This simple policy is shown to be able to perform similarly as sophisticated policy that tracks accesses to cache lines [128].

To quantify the reliability benefit provided by our proposed NV-SRAM based cache with respect to that of drowsy cache, the soft error rate of a drowsy cache cell and a NV-SRAM cell are examined. HSPICE is used to simulate alpha particle striking on both cells, as described in [129]. The methodology reveals how the voltages on the node Q and QB change when a particle with the same energy strikes the cell while it is in the normal and drowsy mode. The simulation results show that the soft error rate increases substantially by 3.2X as the voltage is scaled from 1.1V to 0.3V in the drowsy cache cell. Across all simulated benchmarks, the soft error rate of the drowsy cache is 8.4X greater than that of the NV-SRAM cache.

[130] proposed another approach for reducing cache leakage by invalidating and turning off cache lines when they hold data not likely to be reused. Compared to the proposed scheme, this approach incurs inevitable performance degradation, because reloading data from a lower memory hierarchy level has a significant impact on the performance, especially for the last level
cache. Moreover, the power consumption of reloading data may offset the power saved by
turning off cache lines. In contrast, the proposed scheme allows cache lines to be completely
powered off with data preserved in non-volatile devices at the same cache hierarchy level and the
data restore is nearly instant, thereby introducing negligible performance overhead.

Utilizing NV-SRAM to Reduce Pre-fetching-induced Cache Pollution

Data prefetching is an effective mechanism to hide the long memory access latency. If
prefetch requests are accurate and data are fetched timely, prefetching can improve performance.
Otherwise, it may degrade the overall system performance by polluting the cache. As the non-
volatile elements in NV-SRAM cells provide additional storage capacity, this work proposes to
utilize this capacity to reduce cache pollution due to aggressive or useless pre-fetching.

By utilizing the non-volatile storage, cache lines that are replaced by the pre-fetched cache
lines can be backed up in the non-volatile elements rather than being evicted to the main
memory. Thus, these cache lines can be restored nearly instantly to SRAM cells when they are
access again at a later time instead of reloading from the memory. Figure 6-8 illustrate two cases
in which NV-SRAM’s additional capability can be used to maximize the benefit of the useful
pre-fetches and minimize the negative impact caused by the harmful pre-fetches. In Figure 6-8,
processor computation, including memory references satisfied within the cache hierarchy is
represented by the upper time line, while main memory access is represented by the lower time
line. Figure 6-8(A) shows the case in which NV-SRAM reduces cache pollution caused by
useless or premature pre-fetches. When a hit on cache line X takes place, a pre-fetch of cache
line P is issued and this pre-fetch proceeds in parallel with processor computation. Upon the
completion of memory access, the cache line A residing at the least recently used position is
evicted to make room for the pre-fetched data in a conventional cache. Whereas, the pre-fetch of
cache line P may be a harmful pre-fetch due to the inaccurate pre-fetch prediction or it is
initiated too early. This prematurely pre-fetched data block displaces cache line A that might be
referenced again and results in a cache miss, as is shown in Figure 7-8(a) for the SRAM-based
cache. In contrast, the cache line A is backed up to the non-volatile elements in the NV-SRAM
based cache via a STORE operation and the pre-fetched cache line P is loaded into the SRAM
portion of the cells. Instead of fetching cache line A from the memory, a RECALL operation is
invoked to evict the pre-fetched data in the SRAM portion of the cells and restore the cache line
A back to the SRAM cells, as is shown in Figure 6-8(A) for the NV-SRAM based cache.
Therefore, a cache miss caused by cache pollution can be eliminated. Figure 6-8(B) shows a
scenario that our cache design can further maximize the benefit of useful pre-fetches. The case
shown in Figure 6-8(B) is similar to that in Figure 6-8(A), except that the pre-fetch of cache line
P is a useful pre-fetch and the displaced cache line A is referenced again after the pre-fetched
block has been used. The cache miss on data block A is an ordinary replacement miss for the
conventional cache design since the resulting cache miss would have occurred with or without
pre-fetching. Whereas, this cache miss can be avoided by retaining the displaced cache line A in
the non-volatile storage and restoring it back to SRAM cells in our cache design, thereby
improving the performance.

To exploit the use of non-volatile elements as storage capacity, each cache line requires
two tags to specify the identity of data in the SRAM portion of the cells and in the non-volatile
elements respectively. One approach is to decouple the cache access, adding a level of
indirection between the address tag and the data storage. A decoupled design similar to the one
proposed in [131] is adopted in this work, as is shown in Figure 6-9. The SRAM tag identifies
the data in the SRAM portion of the NV-SRAM cells while non-volatile tag identifies the data in
the non-volatile storage portion. An extra location bit is associated with each address tag to
indicate where the cache line data is stored. A separate cache state indicates the line’s coherence state, which can be M (modified), S (shared), I (invalid), or NP (not present). This decoupled design adds small storage overhead. For example, considering a 16-way, 4MB cache with 64-byte line size, each cache line consists of a 24-bit tag, a 4-bit LRU state, a 2-bit permission information and a data array of 512 bits. This decoupled design adds 2+1+24+1=28 bits per cache line, which introduces only 5% (28/512=5%) additional area overhead.

Stream buffer prefetching is proposed in [132] to store pre-fetched data in a dedicated buffer to overcome the cache pollution problem. However, this solution requires extra latencies in accessing the pre-fetched data in the stream buffer and limits the benefits of accurate pre-fetches. With the proposed technique, the pre-fetched data are allowed to be placed directly within the cache without additional performance penalty upon accurate pre-fetches.

**Achieving Versatile Functionalities Simultaneously through Page Coloring-based Cache Management**

In previous sections, it has been shown that the NV-SRAM cache architecture is able to provide versatile functionalities separately. To allow these features to be available simultaneously, we propose architecture and operating system support to achieve simultaneous versatility. The proposed solution consists of two components: an OS-level page coloring mechanism for cache partition, and a hardware mechanism that allows the operating mode of each NV-SRAM cache partition to be configured independently. Page coloring [133] is a software scheme that manages shared cache space in multi-core platforms to achieve utilization efficiency and fairness by restricting the memory references of an application to a sub-region of the cache. This is achieved by manipulating the mapping between virtual and physical addresses of applications and controlling the mapping of physical memory pages to a processor’s cache.

With this cache partition, different features can be enabled at different regions of the NV-SRAM
cache, depending on which aspect of the application accessing this partition needs to be optimized.

A physical address contains several bits that are common between the cache index and the physical page number. For example as shown in Figure 6-10, for an 8way-associative L2 cache of 4MB with 64 byte cache line, bits 6-18 are used for indexing a cache set and bits 11-31 represent page frame number, assuming a page size of 4K. Note that the upper 7 bits of the cache set index field overlap with the lower 7 bits of the page frame number. Consequently, the value of upper 7 bits of the cache set index can be determined by the OS. As a result, the cache space can be partitioned into a maximum of 128(2^7 = 128) partitions with 64(2^6 = 64) cache sets per partition. Physical memory pages that map to the same partition are assigned to the same color. By specifying the color of pages allocated to an application, the operating system can regulate the cache space usage of an application to one or multiple cache partitions.

Built upon the page coloring-based cache partition, the key of supporting simultaneous versatile functionalities is to allow each cache partition to be configured independently to offer different features at the same time. This is referred as intra-partition-based simultaneous versatility. To achieve this goal, a hardware design is proposed, as illustrated in Figure 6-11. A 3-bit programmable functionality mask register, func_mask_register, is associated with each cache partition. It contains low_cache_interference, low_power, and improved_prefetch bit, which indicate whether the functionality of reducing context-switching cache interference, reducing cache leakage power, and improving prefetch efficiency are enabled or not respectively. The func_mask_register works similarly to an interrupt mask register. The STORE and RECALL requests generated by each NV-SRAM feature are first passed through the func_mask_register to check if they are masked or not. When the corresponding functionality is
disabled (e.g. low power and improved prefetching functionalities are disabled in the Figure 6-11), the STORE and RECALL requests will be masked and consequently they aren’t processed any further. Otherwise (e.g. the functionality of reducing cache interference miss is enabled in the Figure 6-11), these requests will be sent to the peripheral circuits of all cache lines in the same partition to invoke appropriate operations. As a result, programming the

`func_mask_regists` with different values allows the versatile functionalities to be offered across different regions of the cache. Furthermore, the `func_mask_register` can have one or multiple bit set. The latter represents the case in which multiple functionalities are enabled simultaneously within a single NV-SRAM cache partition. This can be achieved by sharing the non-volatile storage across all functionalities. This is referred as `inter-partition-based simultaneous versatility`. Figure 6-12(A) and 6-12(B) shows two cases in which improving prefetching efficiency can be achieved along with reduced cache interference misses and low stand-by power respectively. As shown in Figure 6-12(A), the non-volatile storage may not be occupied entirely for storing its cache data, when process A is swapped out. Thus, process B can store cache lines which are evicted by prefetched data in the non-volatile storage regions which contain no valid data. In Figure 6-12(B), SRAM cells are powered down for saving leakage power. When a prefetched cache line needs to be placed into the cache, the prefetched data can be loaded into SRAM cells after waking them up or override clean cache lines, which operate in the full-power mode. No cache line data is lost during this process and therefore no performance penalty is incurred, since a copy of all cache lines is preserved in the non-volatile storage. Nonetheless, the benefit of NV-SRAM’s low leakage power is sacrificed for achieving better performance due to prefetching. Note that reducing cache interferences is typically not enabled at the same time with minimizing cache leakage power, because it is observed that both of these features impose a high
demand on the non-volatile storage space. In contrast, the feature of improving prefetching efficiency requires less non-volatile storage and thereby can be enabled along with either of the other functionalities.

With the architecture and OS support of simultaneous versatility, a control policy is necessary to determine which functionalities should be enabled on each partition. Such control policy can be either static based on pre-defined configuration or adaptive based on run-time application characteristics. For the static control policy, \texttt{func\_mask\_registers} needs to be programmed appropriately, depending on which aspect of the application needs to be optimized by users and which cache partitions are accessible for this application. For the adaptive control policy, we use the average reuse probability \((P_{\text{avg\_reuse}})\) described in the previous section as an indicator of an application’s run-time characteristics to guide the decision-making process. A low value of \(P_{\text{avg\_reuse}}\) indicates that a small fraction of cache lines are accessed and reused. The rationale of this are twofold: 1) few cache lines are waked up after power down, thereby maximizing the leakage power saving and minimizing the negative performance impact of cache line wake-up; and 2) fewer cache lines are reaccessed when the application resumes execution after context switching, leading to fewer cache interference misses. Therefore, the low-power feature of NV-SRAM is enabled on cache partitions which are accessed by applications with a low \(P_{\text{avg\_reuse}}\) and the low cache interference feature is configured on cache partitions which are referenced by applications with a high \(P_{\text{avg\_reuse}}\). As improving prefetching efficiency can be performed along with either of these features, it is enabled across all cache partition regions by the dynamic control policy.

To implement the proposed architecture and operating system support, a small hardware and software overhead is introduced. The page coloring scheme can be implemented by
extending existing OS default memory allocation scheme to selectively choosing the color of pages to be allocated for an application upon a page fault. The latency of page allocation procedure largely remains the same. The $P_{\text{avg}\_\text{reuse}}$ associated with each process is added into the process descriptor data structure of the kernel. The calculation of $P_{\text{avg}\_\text{reuse}}$ is performed by software routine whenever a context switch occurs and it is conservatively estimated to increase the context switch overhead by 4%. For the hardware support, the 3 bit programmable functionality mask registers and the peripheral logic circuit associated with each cache partition are the only extra hardware overhead introduced, requiring a total of 1920 transistors for a maximum of 128 cache partitions. Both the performance and power overhead are accounted in the experiments.

**Experimental Methodology**

This section describes the experimental methodology for evaluating the benefits of NV-SRAM based cache architecture. The architecture parameters of the simulated microprocessor are listed in Table 6-2. The simulated memory hierarchy consists of two-level caches and an off-chip 2GB DRAM memory. Both L1 instruction cache and L1 data cache are conventional SRAM based caches with a 64KB size and 4-way set associativity. The L2 cache is a NV-SRAM based, 4MB, 16-way associative cache. All caches have the block size of 64 bytes. It is assumed that a 65nm technology with a supply voltage of 1.1V is employed. For performance and power evaluation, a framework is developed based on a heavily extended full-system simulator, PTLSim/X [98], integrated with Wattch power model and the leakage model extracted from HotLeakage [134] and a memory model from DRAMsim [60]. PTLSim/X is a cycle-accurate full system simulator supporting the x86 instruction set architecture and is used to run all of our benchmarks on a Linux kernel 2.6.22.6. PTLSim/X is extended to model NV-SRAM based cache and to estimate its power consumption. For software error vulnerability analysis, the
reliability estimation model developed in [129] is integrated into the simulation framework. The timing and power parameters (listed in Table 6-1) are used in our study. The simulations use all benchmarks from SPEC2000 suite with reference input data set and NAS Parallel Benchmark Version 3.2 with Class “S” input data set. These applications are listed in Table 6-3 along with Miss Per Kilo Instruction (MPKI). All benchmarks are compiled on x86 platform using GCC or FORTRAN compiler with optimization level –O3. For all experiments, the initialization phase is skipped and the detailed cycle-accurate simulation starts at the top of benchmark’s main loop. The simulations stop until at least 1 Billion instructions are committed. The results are reported as an average value over five measurements.

To evaluate context switch overhead due to cache interference, four synthetic programs are created (i.e. stream1MB, stream2MB, stream3MB, stream4MB) and each of them is executed with a benchmark on a time-sharing basis. Each synthetic program has a working set of 1MB, 2MB, 3MB or 4MB and performs sequential access to the entire working set repeatedly. These data set sizes are chosen such that 25%, 50%, 75% or 100% of data in a 4MB cache belonging to the benchmark are evicted before the synthetic program is switched out and the benchmark is switched in. To analyze the sensitivity of each application to the new features offered by NV-SRAM cache, each application is simulated by time-sharing a uniprocessor with the synthetic programs and NV-SRAM based L2 cache is configured to offer reduced cache interference and low leakage separately while always enabling the functionality of improved prefetching. The analysis results are shown in Table 6-3. All workloads are categorized into four classes: C (applications benefit significantly more from reduced cache interference than low power), L (applications benefit significantly more from low power than reduced cache interference), C & L (applications benefit equally from both), N/A (applications don’t benefit from either). To from
workloads consisting of all real-world applications, 4-threaded workloads are created as shown in Table 6-4. The workloads, 2C2L_1, 2C2L_2, 2C2L_3, and 2C2L_4, are formed by using two benchmarks from categories C and L respectively. Workloads 4C&L consist of four benchmarks exclusively from the C&L category. PTLSim/X is used to run these applications on a modified Linux kernel that has no external interference (e.g. various interrupts other than timer interrupts) and measure the number of cycles when the benchmark is executing so that the performance impact of cache interference misses due to context switching is taken into account. To implement the software support for NV-SRAM cache, Linux kernel process scheduling is extended to notify the cache controller for issuing a STORE/RECALL operation upon a context switch involving the benchmark process. To implement page coloring scheme, Linux kernel is modified to extend the memory management module and a policy control management module is added. For the evaluation of pre-fetching, the widely used class of prefetching algorithm know as sequential prefetching is adopted. A simple hardware sequential prefetch engine is implemented, which is similar to the one used in Power5 microprocessor [135][136]. The pre-fetch engine has 12 filters and supports 12 independent streams of sequential prefetch. A pre-fetch request is issued if the predicted address is accessed and up to 8 cache lines ahead of the currently accessed line are pre-fetched.

**Results**

This section presents the performance, power, and reliability benefits of NV-SRAM base cache architecture. The experimental evaluation includes: 1) context switch overhead reduction; 2) power savings achieved by powering off cache lines; 3) reliability improvement due to the immunity of non-volatile devices to soft error strikes; 4) performance improvement due to the reduced cache pollution in prefetching; and 5) simultaneous cache optimization across performance, power, and reliability domains.
Performance Improvement due to the Reduced Context-switch Induced Interferences

Figure 6-13 shows the performance benefits when the NV-SRAM based cache is used to reduce cache interference misses. The case in which a benchmark runs alone on a single processor with a 4MB SRAM based cache is used as our baseline. All other results are normalized with respect to this baseline case. The “stream1MB”, “stream2MB”, “stream3B” and “stream4MB” represent the cases in which each benchmark time-shares a single processor with the synthetic program stream1MB, stream2MB, stream3MB and stream4MB respectively. As shown in Figure 6-13, the performance improvement increases from a geometric mean of 1% to 9% for SPEC 2000 benchmarks and 6% to 21% for NPB benchmark, when the application’s working set displaced by the synthetic program increased from 1MB to 4MB. In the figure, the applications on the x-axis are sorted based on their L2 cache miss rate when they run alone on a single processor. Although the amount of cache data displaced by each of the synthetic programs is roughly the same across all benchmarks, the performance improvement varies from one to another. Thus, there is not an apparent correlation between the cache miss rate and the cache interference misses caused by context switching. For example, despite having similar miss rates, the cache interference misses saved by NV-SRAM based cache allows a 28% performance improvement on vpr, but only 2% on equake, when they execute concurrently with stream4MB. The reason is that not all the displaced data of a process will be accessed again by the process when it resumes execution. Hence, the temporal reuse pattern has a significant impact on how many cache interference misses an application suffers from and consequently affects the maximal performance benefits that NV-SRAM based cache can provide. Compared to NPB benchmarks, SPEC 2000 applications obtain considerably less performance improvement. The analysis on working set and reuse characteristics shows that, for most of SPEC benchmarks, the fraction of unique cache line accessed during a period of time is relatively small, i.e. less than
15% of total cache lines on average. Therefore, a significant fraction of cache lines are not reused after initial accesses and these benchmarks barely benefit from the proposed cache design. In contrast, NPB benchmarks are likely to reuse a large fraction of their working set. This is especially the case on cg, which achieves a 2.8X speedup when it is executed concurrently with stream4MB. Overall, our cache design increases the performance by 15% in the geometric mean across all simulated benchmarks in the best-case scenario (i.e. “stream4MB”).

Figure 6-14 shows the performance improvement when QoS-aware scheme is employed on multiprogramming workloads and compares it with alternative schemes for mitigating context-switching induced cache misses. The “NV-SRAM w/ QoS” case uses NV-SRAM with QoS-aware scheme to dynamically identify the process that is most likely to benefit from the proposed cache architecture and to allow that process to take advantage of non-volatile storage. The “SRAM + Victim Cache” is similar to the baseline, except that an extra victim cache is added, which is exclusively accessed by one application. For the purpose of fair comparison, the victim cache is assumed to be 512KB, so that the total die area of “SRAM + Victim Cache” is approximately the same as that of “NV-SRAM w/ QoS”. The “L1SRAM + L2MRAM” is a hybrid cache design using disparate memory technology proposed in [117]. In this design, L1 is a SRAM cache, while L2 is built from MRAM, which is shown to achieve the best performance in [117]. Note that MRAM is 4X as dense as SRAM, therefore the size of L2 MRAM is 4X as large as “NV-SRAM w/ QoS”, but the read and write latencies of L2 MRAM are significantly larger than that of NV-SRAM based cache and we conservatively assume 20 cycles read latency and 60 cycle write latency[117]. The experimental results (Figure 6-14(A)) show that “NV-SRAM w/ QoS” improves performance by 11% over the baseline, while “SRAM + Victim Cache” only achieves 2% performance enhancement. This is due to the small capacity of victim
cache and the latency in fetching cache data sequentially from it. On the contrary, “L1 SRAM +
L2 MRAM Cache” incurs 12% performance degradation due to the long latency in each L2
access, which completely offsets the benefit of its large capacity. Figure 6-14(B) shows the
breakdown of the percentage of time that each application is given access to the non-voltaile
storage by our QoS-aware scheme in each workload. Note that most of the time the non-volatile
storage is allocated to cg (in 2C2L_1), art (in 2C2L_2), sp and eq (in 2C2L_3), lu (in 2C2L_4),
parser (in 4C&L_1), and bzip (in 4C&L_2). This observation is consistent with the results shown
in Figure 6-13, which suggests that these benchmarks can benefit from the NV-SRAM based
cache substantially.

Power Saving and Reliability Enhancement

Firstly, the energy consumption of the proposed low power cache design is compared with
drowsy cache in Figure 6-15. The results reported in the figure are the normalized total cache
energy over that of conventional SRAM cache. Note that the total cache energy accounts for the
leakage, dynamic energy and the extra energy required to wake up or power down cache lines for
NV-SRAM cache. As can be seen, the NV-SRAM cache design can reduce the L2 cache energy
by 67%, which is 4% more than the 63% achieved by drowsy cache. This energy saving is
primarily contributed by leakage reduction of cache lines in the power-down state. As the
performance penalty of wake-up are very small for both drowsy cache and NV-SRAM based
cache, it is observed that the performance degradation is less than 2% for both cache designs.
Although NV-SRAM cache design takes advantage of the energy benefits of completely turning
off cache lines, the 4% further energy reduction over drowsy cache is much less than the
theoretical maximum of 16%, which is the percentage of leakage energy consumed by a bit in
the drowsy mode with respect to that in the wake-up mode. This is because the operation of
backing up data in the non-volatile elements consume a high energy, which is 7X as much as that
of a cache line access, and therefore it offsets the energy saving benefit. Whereas, it is observed that most cache lines were written to the non-volatile storage only once per period, since unmodified cache lines are not required to update their contents in the non-volatile elements and the simulations show that on average less than 7% of cache lines are updated frequently. Figure 6-16 shows the normalized reliability improvement achieved by using non-volatile devices to preserve data in a low power mode with respect to the baseline case (conventional cache design). Soft Error Rate (SER) is used as the reliability metric, which accounts for both device level failure rate and architecture vulnerability factor. As can be seen, the drowsy cache increases the SER by a geometric mean of 2.3X, which substantially raises the probability of a single event upset in the cache. Hence, drowsy cache achieves the low-power benefit at the cost of the degraded reliability. Whereas, NV-SRAM cache design improves the reliability by 3.1X and 7.1X over the baseline and drowsy cache respectively, allowing a simultaneously optimizations in both energy and reliability without sacrificing one or the other.

**Performance Improvement on Pre-fetching**

To evaluate the efficiency of pre-fetching, 6 memory intensive workloads from SPEC2000 suite that have the highest MPKI values and all 10 benchmarks from NPB suite are used for simulation. Figure 6-17(A) shows the performance speed-up when pre-fetching is applied on the conventional SRAM-based cache and our proposed NV-SRAM based cache and Figure 6-17(B) shows its impact on cache energy. The results are normalized to the case on which no pre-fetching is used. As can be seen in Figure 6-17(A), pre-fetching improves the performance by 8% (geometric mean) when our cache design is adopted, while only 4% performance gain is obtained on conventional SRAM-based cache. The performance improvement is primarily contributed by the reduction of pre-fetching induced cache pollution (the case shown in Figure 6-8(A)). By analyzing the breakdown of performance gain, it is observed that eliminating most of
useless or too early pre-fetches contributes to 74% of this performance improvement. The proposed cache design can effectively mitigate the effect of bad pre-fetches and allow more aggressive pre-fetching to be used for alleviating memory bottleneck. In Figure 6-17(B), it can be seen that the performance enhancement allowed by the proposed NV-SRAM cache can still reduce the cache energy by 5%, although the STORE/RECALL operations performed by NV-SRAM requires 7X more emerging than that of READ/WRITE operations in a conventional SRAM cache.

**Simultaneous Cache Optimizations across Performance, Power, and Reliability Domains**

Figure 6-18 compares the efficiency of NV-SRAM’s simultaneous versatile functionalities through page-coloring scheme against the cases in which NV-SRAM is configured to offer a single feature. The metric “Energy Delay\(^2\) Reliability Product” (ED\(^2\)RP) is used to quantify the overall benefit across performance, power, and soft error reliability domains, the lower the value of ED\(^2\)RP, the better. The reported results are normalized to that of SRAM-based cache design with page-coloring scheme employed. As can be seen, the “Simultaneous Versatile Functionalities” design outperforms the “Interference Reduction” and “Leakage Reduction” by a geometrical mean of 1.6X and 1.9X respectively. Among all simulated workloads, the “Interference Reduction” is only slightly better than “Simultaneous Versatile Functionalities” on 2C2L_1. This is because 2C2L_1 contains the application, cg, which benefits substantially in performance from reduced context-switch cache misses, and the ED\(^3\)RP metric favors performance over power and reliability, thereby leading to a low ED\(^3\)RP on 2C2L_1. Across all workloads, “Leakage Reduction” typically outperforms “Interference Reduction”, since NV-SRAM caches allow low leakage power benefit to be obtained along with improvement soft error resilience and therefore achieves optimizations in both power and reliability domains simultaneously, as is illustrated in previous sections. On the contrary, only the performance is
optimized by “Interference Reduction” and the percentage in improvement is much lower than that of energy saving and reliability improvement. Thus the value of ED²RP is typically higher for “Interference Reduction”. By analyzing the run-time operating mode of all cache partitions, it is observed that the adaptive control policy proposed is able to accurately detect the run-time characteristics of applications and enable the feature that provides the most benefit. For workloads 2C2L_1 , 2C2L_2 , 2C2L_3 , and 2C2L_1 , it is found that two cache partitions are enabled for reducing interferences and the other two partitions are configured to provide low leakage power. Moreover, when the application doesn’t exhibit bias towards one feature with respect to the other, the proposed adaptive control policy configures the cache partition dynamically based on run-time phase behavior, thereby achieving a lower ED2RP than the static configuration, as is illustrated by the results of 4C&L_1 and 4C&L_2.

Related Work

In the past, there have been many studies on context switch overhead analysis, low-power cache design and pre-fetching. [137] investigated how cache parameters and application behavior influence the number of context switch misses the application suffers from. [138] characterized context switch misses of an application for various cache parameters and investigated how a potentially optimum scheduling policy could reduce them. [139] measured both direct and indirect context switching overheads through simulation and concluded that indirect context switch overheads due to the cache perturbation effect are much more significant than direct overheads, which is consistent with the conclusion by performing similar experiments on ARM platform in [122]. To reduce leakage power in cache hierarchy, [128][140] proposed a drowsy cache design using voltage scaling technique. [130][141] reduces the leakage power by turn off the power to the memory cell when the cell is set to low-power mode. To improve pre-fetching accuracy and reduce cache pollution, [142] designed a number of hardware-based pre-fetch
pollution filtering mechanism based on history information. [143] presented an adaptive stream
detection that can modulate the aggressiveness of a stream pre-fetch engine. [144] proposed a
pre-fetch aware memory controller to estimate the usefulness of pre-fetch requests and
dynamically adapt its scheduling and buffer management policies. However, these techniques are
designed to perform optimization in a single domain. This work is unique in that we explore a
new type of memory cell design, non-volatile SRAM, to implement the cache such that this
cache is able to provide benefits in multiple domains.

Due to their low-power features, non-volatile memories such as Flash, PCM, MRAM and
FeRAM are being increasing used to build storage systems and on-chip structures. [63] first
proposed integrating a Flash-based disk cache into memory hierarchy of server platforms to save
memory power. [74][75][99] are the first architecture level studies on using phase change
memory to implement main memory. Whereas, the limited write endurance of Flash, PCM and
FRAM technologies prevent them from being used in the cache hierarchy. [76][77] evaluated
performance and power of 3D stacking cache using MRAM. [79] further evaluated a hybrid
cache design, which takes advantage of the best characteristics of each non-volatile memory
technology. This study differs from these works in two ways: 1) the proposed cache structure is
built from non-volatile SRAM which integrates non-volatile elements into the standard SRAM
cells, combining the high speed characteristics of SRAM cell with the non-volatility feature,
while their caches are constructed only from non-volatile memory devices and is likely to suffer
from high access latency. 2) the proposed NV-SRAM cache design allows data to be copied
between SRAM and non-volatile elements in parallel, thereby allowing nearly instant data
transferring and recovery, while their cache design requires cache lines being fetched
sequentially from a lower level memory hierarchy. Note that the proposed design is difficult than
those reported in [145][146] in that it allows different data to be held in SRAM and non-volatile elements at the same time, therefore efficiently doubling the capacity. Moreover we show that this NV-SRAM design can be used to build a versatile cache with enriched functionality.
Table 6-1. Area, performance and power comparison between SRAM and NV-SRAM based caches

<table>
<thead>
<tr>
<th></th>
<th>SRAM based Cache (4MB)</th>
<th>NV-SRAM based Cache (4MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Die area(mm$^2$)</td>
<td>71.7</td>
<td>79.0</td>
</tr>
<tr>
<td>Read latency(ns)</td>
<td>2.4</td>
<td>2.4</td>
</tr>
<tr>
<td>Write latency(ns)</td>
<td>2.3</td>
<td>NORMAL</td>
</tr>
<tr>
<td>Read energy(nJ)</td>
<td>0.9</td>
<td>Mode</td>
</tr>
<tr>
<td>Write energy(nJ)</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>Total leakage(W)</td>
<td>10.7</td>
<td>11.1</td>
</tr>
<tr>
<td>Store latency(ns)</td>
<td></td>
<td>STORE</td>
</tr>
<tr>
<td>Store energy(nJ)</td>
<td></td>
<td>6.2</td>
</tr>
<tr>
<td>Recall latency(ns)</td>
<td></td>
<td>RECALL</td>
</tr>
<tr>
<td>Recall energy(nJ)</td>
<td></td>
<td>1.0*</td>
</tr>
</tbody>
</table>

* The recall latency and energy are lower than those of SRAM read access because the bit and bit_bar lines are not charged or discharged and no sense amplifier is required.

Table 6-2. Baseline machine configuration

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frequency</td>
<td>3G Hz</td>
</tr>
<tr>
<td>Width</td>
<td>4-wide fetch/decode/issue/commit</td>
</tr>
<tr>
<td>IQ</td>
<td>64 entries</td>
</tr>
<tr>
<td>ITLB</td>
<td>128 entries, 4-way</td>
</tr>
<tr>
<td>BranchPred.</td>
<td>2K entries Gshare, 10-bit global history</td>
</tr>
<tr>
<td>BTB</td>
<td>2K entries, 4-way</td>
</tr>
<tr>
<td>RAS</td>
<td>32 entries RAS</td>
</tr>
<tr>
<td>L1 I-Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 2 cycles</td>
</tr>
<tr>
<td>ROB Size</td>
<td>128 entries</td>
</tr>
<tr>
<td>LDQ</td>
<td>48 entries</td>
</tr>
<tr>
<td>STQ</td>
<td>32 entries</td>
</tr>
<tr>
<td>Int ALU</td>
<td>4 I-ALU, 2 I-MUL/DIV, 1 Load/Store</td>
</tr>
<tr>
<td>FP ALU</td>
<td>2 FP-ALU, 2 FP-MUL/DIV/SQRT</td>
</tr>
<tr>
<td>DTLB</td>
<td>256 entries, 4-way</td>
</tr>
<tr>
<td>L1 D-Cache</td>
<td>64KB, 4-way, 64 Byte/line, 2 ports, 2 cycles</td>
</tr>
<tr>
<td>L2 Cache</td>
<td>4MB, 16-way, 64 Byte/line, 8 cycles</td>
</tr>
<tr>
<td>Memory</td>
<td>DRAM, 2GB, 8 banks</td>
</tr>
<tr>
<td>Write Buffer</td>
<td>4 entries, 512B per entry</td>
</tr>
<tr>
<td>Size</td>
<td>2GB</td>
</tr>
<tr>
<td>Row</td>
<td>16384</td>
</tr>
<tr>
<td>Frequency</td>
<td>667MHz</td>
</tr>
<tr>
<td>Number of Banks</td>
<td>8</td>
</tr>
<tr>
<td>Number of Columns</td>
<td>2048</td>
</tr>
<tr>
<td>Data Width</td>
<td>64bit data + 8 bit ECC</td>
</tr>
<tr>
<td>Row Buffer policy</td>
<td>Open page</td>
</tr>
<tr>
<td>Refresh Interval</td>
<td>64ms</td>
</tr>
</tbody>
</table>
Table 6-3. Benchmarks

<table>
<thead>
<tr>
<th>Benchmarks</th>
<th>MPKI (4MB)</th>
<th>Benefit Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>dc</td>
<td>17.1</td>
<td>C</td>
</tr>
<tr>
<td>ft</td>
<td>16.5</td>
<td>C</td>
</tr>
<tr>
<td>is</td>
<td>8.6</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>cg</td>
<td>6.3</td>
<td>C</td>
</tr>
<tr>
<td>ua</td>
<td>6.2</td>
<td>N/A</td>
</tr>
<tr>
<td>mg</td>
<td>6.1</td>
<td>C</td>
</tr>
<tr>
<td>lu</td>
<td>5.2</td>
<td>C</td>
</tr>
<tr>
<td>sp</td>
<td>3.3</td>
<td>C</td>
</tr>
<tr>
<td>bt</td>
<td>3.2</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>ep</td>
<td>3.0</td>
<td>C</td>
</tr>
<tr>
<td>ammp</td>
<td>14.5</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>applu</td>
<td>10.7</td>
<td>L</td>
</tr>
<tr>
<td>mgrid</td>
<td>9.0</td>
<td>L</td>
</tr>
<tr>
<td>apsi</td>
<td>9.0</td>
<td>N/A</td>
</tr>
<tr>
<td>gap</td>
<td>8.6</td>
<td>L</td>
</tr>
<tr>
<td>swim</td>
<td>8.0</td>
<td>L</td>
</tr>
<tr>
<td>mcf</td>
<td>6.6</td>
<td>L</td>
</tr>
<tr>
<td>facerec</td>
<td>5.5</td>
<td>C</td>
</tr>
<tr>
<td>bzip</td>
<td>4.8</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>gzip</td>
<td>4.6</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>art</td>
<td>1.9</td>
<td>C</td>
</tr>
<tr>
<td>parser</td>
<td>1.5</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>vpr</td>
<td>0.9</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>sixtrack</td>
<td>0.9</td>
<td>N/A</td>
</tr>
<tr>
<td>gcc</td>
<td>0.7</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>vortex</td>
<td>0.7</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>lucas</td>
<td>0.6</td>
<td>L</td>
</tr>
<tr>
<td>galgel</td>
<td>0.6</td>
<td>C</td>
</tr>
<tr>
<td>crafty</td>
<td>0.3</td>
<td>C &amp; L</td>
</tr>
<tr>
<td>mesa</td>
<td>0.3</td>
<td>L</td>
</tr>
<tr>
<td>equake</td>
<td>0.3</td>
<td>L</td>
</tr>
<tr>
<td>twolf</td>
<td>0.1</td>
<td>L</td>
</tr>
</tbody>
</table>

C: benefit significantly more from reduced cache interference than low leakage power

L: benefit significantly from low leakage power than reduced cache interferences

C & L: benefit equally from reduced cache interferences and low leakage power

N/A: not benefit from either

Table 6-4. Workloads

<table>
<thead>
<tr>
<th>Type</th>
<th>Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workloads consisting of mix of applications that benefit from either reduced cache interferences or low leakage power</td>
<td>2C2L_1 (cg, galgel, eon, twolf)</td>
</tr>
<tr>
<td>Workloads consisting of applications benefit equally from reduced interferences and low power features</td>
<td>4C&amp;L_1 (vpr, gcc, vortex, parser)</td>
</tr>
</tbody>
</table>

2C2L_2 (art, mg, mesa, perlmbk) 2C2L_3 (sp, ep, equake, lucas) 2C2L_4 (lu, dc, mgrid, applu)
Figure 6-1. (A) Parallel state (state “0”) (B) Anti-parallel state (state “1”).

Figure 6-2. The structure of a NV-SRAM cell.

Figure 6-3. NV-SRAM operation modes (V_{high} : full supply voltage level , V_{low} : zero supply voltage level ).
Figure 6-4. HSPICE simulation waveforms of NV-SRAM RECALL operation.

Figure 7-6. (A) 3D-structure of a NV-SRAM cell (B) Layout of a conventional SRAM cell (C) Layout of a NV-SRAM cell.
Figure 6-6. NV-SRAM based cache architecture.

Figure 6-7. The use of NV-SRAM to reduce cache interference misses.

Figure 6-8. The illustration of cases in which NV-SRAM cache can improvement pre-fetching performance.
Figure 6-9. The tag array of decoupled NV-SRAM Cache.

<table>
<thead>
<tr>
<th>LRU State</th>
<th>Tag SRAM</th>
<th>Tag Nonvolatile</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 bits</td>
<td>1 bit</td>
<td>24 bits</td>
</tr>
</tbody>
</table>

Permission: M (modified), S(shared), I (invalid), NP(not present)

Location: 1 if cache line data are in the SRAM portion of the cells and can be accessed directly, 0 otherwise

Figure 6-10. An illustration of the page coloring technique.

Figure 6-11. Intra-partition-based simultaneous versatility.
Figure 6-12. Inter-partition-based simultaneous versatility.

Figure 6-13. Performance improvement due to the reduced interference.

Figure 6-14. (A) Performance improvement comparison due to the reduced context switch overhead on four-threaded multiprogramming workloads (B) Percentage of time that each application is allowed to use the non-volatile storage by NV-SRAM caches with QoS.
Figure 6-15. A comparison of overall L2 cache energy.

Figure 6-16. A comparison of the soft error rate of L2 cache.

Figure 6-17. (A) The efficiency of pre-fetching on NV-SRAM based caches (B) the entering impact.
Figure 6-18. The efficiency of simultaneous versatile functionalities.
CHAPTER 8
CONCLUSIONS

Understanding the characteristics of emerging technologies and exploring their benefits are crucial for the design of future high-performance, energy-efficient, and reliable microprocessor system. Toward this end, this research has investigated the characteristics of emerging 3D integration technology and new memory technologies in microprocessor and memory system design. Microarchitecture-level and system-level techniques are proposed to explore opportunities provided by these new technologies in optimizing performance, power, and reliability of microprocessor systems.

For the application of 3D integration in microarchitecture design, previous studies have already demonstrated that 3D stacking technologies can provide a substantial performance increase and power reduction. However, the impact of 3D fabrication on the design of reliable microarchitecture is still largely unknown. Through detailed analysis of soft error physical mechanism, this work shows the heterogeneous soft error resilience characteristic of 3D integration’s vertical stacking structure due to its shielding capability. The vertically stacked structure allows outer-layers to shield inner-layers from particle strikes. By taking advantage of the program characteristics, the reliability benefit of 3D’s shielding effect is further exploited by mapping vulnerable instructions to robust layers, and utilizing narrow-width values and ASRAM cell in 3D register file design. The major conclusions that can be drawn from this part of the study are: 1) This research shows that 3D-implemented processor design has a substantial reduction in soft error rate, which is estimated to be 70% in the experimental results. 2) This reliability benefit is of considerable importance due to the potential of reduction or even elimination of the overhead involved by conventional reliability optimization techniques, thereby minimizing the hardware complexity and the cost for achieving the same reliability target. 3)
Although this work shows that the reliability benefit increases as more dies are stacked on top of each other, thermal issues are likely to occur. Thus a trade-off needs to be made so that the strength of 3D integration can be maximized across different domains.

One of the greatest advantages of 3D is its ability to enable die-stacked on-chip memory, which provides a significant relief from the Memory Wall problem. This will likely be pursued by industry as a first step of adopting 3D in real designs. Nevertheless, the elevated on-chip temperature in 3D presents a significant challenge for the power management of DRAM-based die-stacked memory. Moreover, DRAM is facing scalability and large power consumption issues. Emerging phase change memory technology provides a promising solution to address all these issues and is regarded as one of the most promising technologies for the next generation of memory systems. To design PCM-based memory systems, this work analyzes the thermal characteristics of PCM and shows that PCM is high-temperature friendly. This thermal-friendly feature along with low standby power and superior scalability makes PCM and 3D technology fit very well with each other. However, phase change memory has longer read/write access latencies than DRAM and is subject to wear-out, reliability issues, and process variation. To reduce the latency of PCM, 3D vertical interconnection is utilized and a hybrid PCM /DRAM memory architecture with OS support is presented. To enhance the endurance, a scheme of using variable strength error correction code is proposed along with reliability-aware OS paging to achieve ware-leveling. To mitigate the impact of design parameter variations, data comparison write, memory compression scheme, and OS paging technique are proposed. To enhance the reliability of future high-density multi-level PCM, data inversion and rotation, hybrid SLC/MLC design and temperature-aware page allocation scheme are proposed to tolerate resistance drift induced readout errors. These techniques allow smaller margins design to satisfy the reliability
requirements in a power- and endurance- efficient manner. This part of the research suggests the following conclusions: 1) The two emerging technologies, 3D integration and PCM, can fit well with each other. The short vertical interconnection allowed by 3D reduces both access latency and power consumption of PCM. The high-temperature driven operation and low stand-by power of PCM can alleviate the effect of thermal issues in 3D chips. 2) The alleviation of thermal effects allows more aggressive die stacking for maximizing the benefit of 3D technology. 3) Process variation and reliability of single-/multi- level PCM need to be taken into consideration for memory system design using PCM.

The emerging non-volatile memory technologies provide new opportunities not only for memory architecture but also for cache design. In this research, a NV-SRAM cell design that synergically integrates non-volatile devices into the standard SRAM cells is presented. By using detailed circuit-level simulations, this work shows that the proposed NV-SRAM based cache not only has the same high-speed characteristic as that of a conventional SRAM based cache, but also provides versatile and enriched features, such as non-volatility, additional storage capacity and instant data storing and recovering. By taking advantage of these features, this work showcases a wide range of applications in performance and power optimizations. For example, the use of NV-SRAM based cache is proposed to effectively mitigate the performance impact of context-switching induced cache interference misses. It is shown that cache data can be preserved in the non-volatile storage without power supply to reduce leakage power consumption. Moreover, the additional capacity provided by non-volatile devices can be exploited to reduce prefetching-induced cache pollution. This work also presents architecture and operating system support to achieve simultaneous, multiple domain cache optimization by using NV-SRAM based cache architecture. This part of the research suggests that the non-volatile and
non-transistor-based features of emerging memory technologies provide opportunities to enrich the capabilities of future cache and memory architecture design. For example, the persistence feature can provide instantaneous boot/hibernate capability, inexpensive check pointing, stronger safety guarantee in file systems etc. Furthermore, the superior scalability of new memory technologies implies much higher density than DRAM or SRAM, lower cost and substantially increased capacity to accommodate the much larger working set of future chip multi-processor systems.
REFERENCE LIST


[71] Micron DDR2 SDRAM system-Power Calculator.


BIOGRAPHICAL SKETCH

Wangyuan Zhang was born in Wuhu, China in 1983. He graduated from Beihang University, China, in 2005 with a Bachelor of Science degree in electrical engineering. After that, he joined the Department of Electrical and Computer Engineering at the University of Florida. In 2007, he earned his Master of Science degree in electrical and computer engineering. The same year, he joined IBM Austin Research Lab as a graduate research intern and performed research in PowerPC microprocessor design. He is a student member of The Institute of Electrical and Electronics Engineers (IEEE), The Institute of Electrical and Electronics Engineers (IEEE) Computer Society, Association for Computing Machinery (ACM).