ANALYSIS OF
MULTI-MEGABYTE SECONDARY CPU CACHE MEMORIES

by

RICHARD EUGENE KESSLER

A thesis submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy
(Computer Science)

at the
UNIVERSITY OF WISCONSIN-MADISON
1991

© copyright by Richard Eugene Kessler 1991
All Rights Reserved

Analysis of Multi-Megabyte Secondary CPU Cache Memories

Richard Eugene Kessler

This dissertation investigates multi-megabyte secondary caches. In a multi-level cache hierarchy, secondary caches service processor memory references that cannot be serviced by smaller and faster primary caches. With faster processors and expanding main memories, a multi-megabyte cache is increasingly vital because it shields processor memory references from costly main memory accesses, even when the processor references a large address space. Multi-megabyte secondary caches allow processors to execute at the speeds they are capable of, even when there is a large processor-to-main-memory speed gap.

This analysis uses a new collection of memory address traces that is appropriate for multi-megabyte cache simulation. These traces thoroughly characterize several large workloads, and are long enough (billions of instructions) to overcome multi-megabyte cache cold-start. This dissertation includes the first comparison of two previously-proposed trace-sampling techniques that can reduce long-trace simulation requirements: *set sampling* and *time sampling*. Under a range of conditions, set sampling produces more accurate cache performance estimates with less trace data than time sampling.

This dissertation examines many alternative cache designs. It shows that multi-megabyte secondary caches are extremely useful with large processor-to-main-memory speed gaps. Furthermore, associativity is needed for smaller secondary caches, but may not be necessary in multi-megabyte caches; multi-megabyte cache block sizes should be larger than for smaller caches, and the block size that equals the fixed latency and transfer time miss penalty components is a good design point.

Finally, this dissertation introduces and solves the problems caused by the interaction of virtual memory and real-indexed multi-megabyte caches. Since the placement of pages in main memory also places data in the cache, a poor page placement will cause poor cache performance. This dissertation introduces several new *careful page mapping* algorithms to improve the page placement, and shows that they eliminate 10%-20% of the direct-mapped real-indexed cache misses for the long traces. In other words, this dissertation develops software techniques that can make a hardware direct-mapped cache appear about 50% larger.

Acknowledgements

There are so many that helped put this dissertation together that I don't know where to start. Perhaps most importantly, I want thank the National Science Foundation and the University of Wisconsin Alumni Research Foundation for financially supporting my graduate career through fellowships. Digital Equipment also helped financially through an internship during the summer of 1989. My advisor, Mark Hill, was always a source of drive and inspiration. Mark patiently corrected many of the errors in this dissertation. The mistakes that remain in this dissertation, however, are mine. Without him, this thesis would not exist.

I also owe a lot to Anita Borg and David Wall, and the rest of DEC's Western Research Laboratory. They developed the trace-gathering system that I needed to get the quantitative (i.e. trace-driven simulation) results for this dissertation. I am grateful to Joel Bartlett, Renato De Leone, Jeremy Dion, Norm Jouppi, Bob Mayo, and Don Stark for providing traceable applications. Paul Vixie and Colleen Hawk also helped store the traces. Mike Litzkow and Miron Livny were a great help in adapting Condor to the requirements of these simulations. Paul Beebe and the Systems Lab were able to satisfy the *enormous* computing needs of this dissertation.

I thank many others for their useful feedback. David Wood made numerous useful comments. He was closely involved with the trace-sampling work. Guri Sohi made many critical comments that greatly improved this dissertation. I also wish to thank Jim Goodman and Parameswaran Ramanathan for serving on my committee. Special thanks to Harold Stone for his useful comments on the research in this dissertation.

Last, but not least, I am grateful to my brothers, sisters, and parents for helping me keep things in proper perspective. Finally, I will be forever grateful to Lisa Snow, both for helping to find errors, and for just being there.

Table of Contents

Chapter 1. Introduction ... 1

1.1. Dissertation Overview and Contributions ... 2

Chapter 2. Background and Cache Performance Analysis ... 5

2.1. Introduction ... 5
2.2. Virtual Memory Concepts ... 5
2.3. Set-Associative CPU Cache Concepts ... 6
2.4. Previous Cache Analysis ... 9
2.4.1. Hardware Monitoring ... 9
2.4.2. Analytical Modeling ... 10
2.4.3. Trace-Driven Simulation ... 10
2.4.3.1. Gathering Address Traces ... 11
2.4.3.2. Cache Analysis Using Address Traces ... 13
2.5. Cache Performance Metrics ... 14
2.6. Statistical Techniques ... 17
2.7. The Trace-Driven Simulation Environment ... 18
2.8. Conclusions ... 20

Chapter 3. Long Traces For Cache Analysis ... 22

3.1. Introduction ... 22
3.2. The Trace Gathering and Storage Mechanism ... 23
3.2.1. Gathering Long Traces ... 23
3.2.2. Storing Long Traces ... 25
3.3. A Description of the Workloads ... 30
3.4. Simulation Results from the Traces ... 30
3.4.1. Instruction Mix and Memory Usage ... 30
3.4.2. The Multiprogrammed Traces ... 33
3.4.3. Miss Frequencies in Multi-Level Cache Configurations ... 35
3.4.4. Secondary Cache Inter-Miss Distributions ... 39
3.5. The Advantages of Long Traces ... 44
3.5.1. Capturing More Workload Behavior ... 45
3.5.2. Overcoming Cache Initialization ... 48
3.6. Conclusions ... 54

Chapter 4. Trace-Sampling Techniques ... 56

4.1. Introduction ... 56
4.2. Set Sampling ... 58
4.2.1. Obtaining Unbiased Estimators from Set Samples ... 58
4.2.1.1. Constructing the Set Sample ... 58

4.2.1.2. Estimating *MPI* With a Set Sample 61
4.2.2. What Fraction of the Full Trace is Needed? 62
4.2.3. Advantages and Disadvantages of Set Sampling 69
4.3. Time Sampling 69
4.3.1. Obtaining Unbiased Estimators from Time Samples 69
4.3.2. What Fraction of the Full Trace is Needed? 78
4.3.3. Advantages and Disadvantages of Time Sampling 86
4.4. Conclusions 87

Chapter 5. Memory System Design With Multi-Megabyte Caches 89

5.1. Introduction 89
5.2. Importance of Multi-Megabyte Caches 90
5.3. Importance of the CPU Cache Hierarchy 91
5.4. The Primary Caches 93
5.5. Secondary Cache Size Choice 95
5.5.1. Effect of Processor-Main Memory Speed Gap 95
5.5.2. The Effect of Workload Choice 96
5.5.3. Effect of Speed Degradation With Larger Caches 97
5.6. Secondary Cache Associativity Alternatives 99
5.6.1. The Usefulness of Associativity 100
5.6.2. Replacement Policy Effects 102
5.6.3. Effect of Speed Degradation With Associativity 103
5.6.4. Inexpensive Associativity Implementations 106
5.7. Secondary Cache Block Size Alternatives 112
5.8. Inclusion Design Alternatives 115
5.9. Conclusions 118

Chapter 6. Page Placement Algorithms for Real-Indexed Caches 120

6.1. Introduction 120
6.1.1. Previous Work 121
6.1.2. Contributions of this Chapter 122
6.2. Motivation for Careful Page Mapping 124
6.2.1. Page Placement in Cache Bins 123
6.2.2. A Simple Static Page Conflict Analysis 124
6.3. Careful Mapping Implementations 128
6.3.1. Page Coloring 129
6.3.2. Bin Hopping 130
6.3.3. Best Bin 131
6.3.4. Hierarchical 134
6.4. Trace-Driven Simulation Performance Analysis 138
6.4.1. The Simulator 139
6.4.2. Results 140
6.4.2.1. The Usefulness of Careful Page Mapping 140
6.4.2.2. Static Page Conflict Minimization 145
6.4.2.3. Careful Mapping Alternatives 147
6.4.2.4. Does PID-Hashed Virtual-Indexing Approximate Real-Indexing? 148

6.4.2.5. Page Size Performance Effects 150
6.4.2.6. The Effect of Main Memory Contention 151
6.5. Discussion 153
6.6. Conclusions 154

Chapter 7. Summary and Future Work 156

7.1. Summary 156
7.2. Future Work 158

Appendix A. 160

Appendix B. 164

Bibliography. 168

List of Figures

Chapter 1.

1.1. Hierarchical Memory System Evolution 3

Chapter 2.

2.1. Virtual to Real Address Translation 7
2.2. Set-Associative CPU Cache Indexing 9
2.3. Trace-Driven Simulation 11
2.4. Cache Performance Metrics 15
2.5. Simulation of Homogeneous Hierarchical Configurations 19
2.6. Parallel Simulation with Condor 20

Chapter 3.

3.1. Code Expansion For Tracing 24
3.2. Concatenating Multiple Trace Buffers 25
3.3. The Phases of Trace Compression 26
3.4. Basic Block Compression Code 27
3.5. Basic Block Decompression Code 28
3.6. Mult2.2 Processes Classified by Execution Length 35
3.7. Mult2.2 Processes Classified by Memory Usage 36
3.8. Miss Reduction for Doubling Associativity 40
3.9. Miss Reduction for Doubling Size 41
3.10a. Inter-Miss Density of Mult1, Mult1.2, and Mult2 42
3.10b. Inter-Miss Density of Mult2.2, Tv, and Sor 43
3.10c. Inter-Miss Density of Tree and Lin 44
3.11a. Variability of Mult1, Mult1.2, and Mult2 46
3.11b. Variability of Mult2.2, Tv, and Sor 47
3.11c. Variability of Tree and Lin 48
3.12. Fraction of Misses that are Compulsory 49
3.13. Cumulative Initialization References for Beginning of Mult1.2 52
3.14. Memory Referenced Over Length of Trace 53

Chapter 4.

4.1. Sampling as Vertical and Horizontal Time-Space Slices 57
4.2. Set Sampling on the Mult1.2 Trace 63
4.3. Distribution of Set Samples from the Mult1.2 Trace 64
4.4. Range of Performance With Set-Samples 67
4.5. Accuracy of Time Sampling with Mult1.2 80
4.6a. Accuracy of Time Sampling with Mult1, Mult2, Mult2.2, and Tree 82
4.6b. Accuracy of Time Sampling with Tv, Sor, and Lin 83

4.7. Accuracy of Time Sampling for Different Cache Sizes 84
4.8. Range of Time-Sampling Estimate for Mult1.2 85
4.9. Time-Sample Distributions.. 86

Chapter 5.

5.1. Memory Traffic Reduction With Multi-Megabyte Cache 91
5.2. The Simulated Cache Configuration.. 92
5.3. Portions of SCPI for Mult1.2 94
5.4. Effect of Miss Penalty on SCPI for Mult1.2.. 96
5.5. Effect of Workload on Cache Size Choice 97
5.6. Effect of 10% Speed Degradation Per Cache Size Doubling 98
5.7. Break-Even Access Time Penalties for Cache Size Doubling 100
5.8. Performance Over a Range of Associativities 101
5.9. Associativity Performance With 10% Speed Degradation 105
5.10a. Direct-Mapped to 2-Way Break-Even Implementation Penalties 106
5.10b. Direct-Mapped to 2-Way Break-Even (Blow-up) 107
5.11. Implementing Set-Associativity 108
5.12. Improved Implementations of Serial Set-Associativity.. 109
5.13. Performance Over a Range of Block Sizes... 114

Chapter 6.

6.1. Virtual Memory and Set Indexing Interaction.. 121
6.2. A Virtual-Indexed Bin Placement.. 123
6.3. Random and Careful Real-Indexed Bin Placements... 125
6.4. Cache Placement Conflicts 127
6.5. Random Mapping Conflicts.. 128
6.6. Bin Choice Code for Page Coloring 130
6.7. Page Placement By Bin Hopping... 131
6.8. Bin Choice Code for Bin Hopping 132
6.9. Code to Choose Among Bins... 133
6.10. Bin Choice Code for Best Bin 133
6.11. Page Placement by Best Bin 134
6.12. Page Placement Using Hierarchy... 136
6.13. Bin Tree Traversal 135
6.14. Mapping Tree Nodes to Bins 137
6.15. Bin Choice Code for Hierarchical 138
6.16. LRU List Implementation With Available Page Frames... 139
6.17. Mult2.2 Results for Different Page Mappings... 141
6.18. Direct-Mapped Hierarchical Miss Reductions for All Traces 142
6.19. Mult2.2 Careful Mapping Improvement.. 144
6.20. Mult1.2 Results for Different Page Mappings... 144
6.21. Conflicts for Mult2.2... 146
6.22. Hierarchical Reduction vs. Conflicts Eliminated... 147
6.23. Virtual and Real Indexing Comparison 150
6.24. Virtual and Real Indexing Associativity Improvement 151

List of Tables

Chapter 3.

3.1. The Types of Preprocessed Trace Entries 29

3.2. A Description of the Traced Programs 31

3.3. A Description of the Studied Workloads 32

3.4. Instruction Information on the Traces 33

3.5. Memory Requirements of Traces 33

3.6. Process Statistics of Multiprogrammed Traces 34

3.7. Primary Cache MPI 36

3.8. Secondary Cache MPI 38

3.9. Instructions to Fill Primary Caches 49

3.10. Instructions to Fill Direct-Mapped Secondary Caches 50

3.11. Instructions to Fill 4-way Secondary Caches 51

3.12. Instructions to Warm Cache to Knee 52

3.13. Trace Length for Five Misses Per Block Frame 54

Chapter 4.

4.1. Errors of Different Set-Sample MPI Estimates 62

4.2. Set Sampling Coefficients of Variation for Direct Mapped 65

4.3. Set Sampling Coefficients of Variation for 2-Way 66

4.4. Set-Sampling Error Prediction 68

4.5. Accuracy of Cold-Start Techniques for Mult1.2 72

4.6. Accuracy of Cold-Start Techniques With Direct-Mapped Caches 73

4.7. Accuracy of Cold-Start Techniques With 4-Way Set-Associativity 74

4.8. Scoring of Different Cold-Start Techniques 75

4.9. μ and the Accuracy of its Estimator ($\hat{\mu}_{split}$) 76

4.10. Accuracy of INITMR Time-Sample MPI Estimates 77

4.11. Individual Versus Merged $\hat{\mu}_{split}$ Estimation 79

Chapter 5.

5.1. Primary Cache Performance 94

5.2. Effects of Associativity on Cache Performance 102

5.3. Improvement of Replacement Policies 104

5.4. Expected Probes of Associativity Implementations 110

5.5. Read Probes of Inexpensive Associativity Implementations 111

5.6. Block Size Alternatives 115

5.7. Implicit and Explicit Inclusion Invalidate Frequency 117

Chapter 6.

6.1. Maximum Conflict Fraction Approximations 129

6.2. Careful Mapping Comparison 148
6.3. Performance of Mapping Schemes for Mult2.2... 149
6.4. Cache Misses vs. Page Size 151
6.5. Memory Referenced vs. Page Size 151
6.7. Cache and Page Misses for Memory Contention.. 153

List of Definitions

Chapter 2.

2.1. Definitions for Virtual Memory 5
2.2. Definitions for Caches 7

Chapter 4.

4.1. Set Sample 59
4.2. Constant-Bits Sample 59

Appendix A.

A.1. Page Mapping Function 160
A.2. Page Frame and Cache Bin Correspondence 160
A.3. Cache Size Independent Mapping Function 161
A.4. Bit-Reversed Bin Tree 161

List of Theorems

Chapter 4.

4.1. When a Constant-Bits Sample is a Set Sample 59

Appendix A.

A.1. Value of BRBT Node.. 161
A.2. Size Independence of Hierarchical 162