



# **An Experimental System for Evaluating Cache Coherence Protocols in Shared Memory Multiprocessors**

Peter John Ashenden

Department of Computer Science  
The University of Adelaide

Submitted for the Degree of Doctor of Philosophy  
in  
January 1997  
Revised August 1997

© 1997, Peter J. Ashenden

# Contents

## Chapter 1

<b>Introduction</b> .....	<b>1</b>
1.1 Workstations and Networks .....	1
1.2 Multiprocessor Architectures .....	2
1.3 Bus Connected Shared Memory Multiprocessors .....	4
1.4 Contribution .....	7

## Chapter 2

<b>Cache Coherence and Performance Evaluation</b> .....	<b>10</b>
2.1 Caches and Cache Coherence .....	10
2.2 Survey of Cache Coherence Protocols .....	14
2.2.1 Protocol Terminology .....	14
2.2.2 Goodman's Write-once Protocol .....	17
2.2.3 The Illinois Protocol .....	21
2.2.4 The Synapse Protocol .....	26
2.2.5 The Berkeley Ownership Protocol .....	30
2.2.6 The Sun MBus Protocol .....	35
2.2.7 The Dragon Protocol .....	39
2.2.8 The Original Firefly Protocol .....	44
2.2.9 The Published Firefly Protocol .....	48
2.2.10 The RB and RWB Protocols .....	53
2.2.11 Correctness of Coherence Protocols .....	54
2.3 Protocols Used by Current Processors .....	56
2.3.1 Intel Pentium® .....	56

2.3.2	IBM PowerPC .....	57
2.3.3	Sun Microsystems UltraSPARC™ -II .....	58
2.3.4	MIPS R4000 .....	58
2.3.5	DEC Alpha .....	58
2.4	Proposed Futurebus Cache Coherence Mechanisms .....	61
2.4.1	P896.2 Signalling Mechanisms .....	62
2.4.2	P896.2 Cache Coherence Rules .....	65
2.4.3	Using P896.2 to Implement the Berkeley Protocol ...	70
2.4.4	Using P896.2 to Implement the Dragon Protocol ....	74
2.4.5	Summary of P896.2 Options .....	79
2.4.6	Correctness and Completeness of the P896.2 Rules ..	82
2.5	Performance Evaluation of Coherence Protocols .....	84
2.5.1	Analytical Evaluation .....	86
2.5.2	Simulation Based Evaluation .....	91
2.5.3	Evaluation Using Real Systems .....	93
2.6	Summary .....	94

### Chapter 3

<b>The Leopard Multiprocessor .....</b>	<b>96</b>
3.1 Background .....	96
3.2 Leopard Architectural Framework .....	97
3.3 The Leopard-1 Multiprocessor .....	99
3.4 The Leopard-2 Multiprocessor .....	101
3.5 The Leopard-2 System Bus .....	104
3.5.1 Arbitration Protocol .....	105
3.5.2 Data Transfer Protocol .....	106
3.5.3 System Maintenance .....	109
3.6 The Leopard-2 General Data Processor .....	112
3.7 The L2GDP Programmable Cache Design .....	114

3.7.1	Cache Organization .....	114
3.7.2	Cache Data Paths .....	116
3.8	Cache Operation .....	121
3.8.1	CPU Requirements of the External Cache .....	121
3.8.2	CPU Cachable Read and Write Requests .....	123
3.8.3	CPU Flush .....	132
3.8.4	Snoop Operation .....	133
3.8.5	Asynchronous Writes from the Write Buffer .....	138
3.9	Summary .....	140
<b>Chapter 4</b>		
	<b>A Programmable Cache Controller for the Leopard-2 .....</b>	<b>141</b>
4.1	Introduction .....	141
4.2	Cache Controller Configuration Parameters .....	142
4.3	A VHDL Model of the Programmable Cache Controller .....	145
4.3.1	The Leopard-2 System Model .....	145
4.3.2	Workload Modelling in the Processor Block .....	148
4.3.3	The Cache Model .....	152
4.3.4	The Coherence Monitor Model .....	187
4.4	Summary .....	189
<b>Chapter 5</b>		
	<b>Conclusions .....</b>	<b>191</b>
5.1	Summary of Project Context .....	191
5.2	Experimental Evaluation of Cache Coherence Protocols .....	193
5.3	Conclusion .....	195
<b>Appendix A</b>		
	<b>L-Bus Data Transfer Protocol .....</b>	<b>196</b>
A.1	Overview .....	196

A.2	Addressing Structure .....	197
A.3	Data Transfer Signals .....	199
A.3.1	Information Signals .....	199
A.3.2	Master Command Signals .....	199
A.3.3	Cache Status Signals .....	200
A.3.4	Slave Status Signals .....	200
A.3.5	Sequencing Signals .....	201
A.3.6	Slot Address .....	202
A.4	Data Transfer Protocol Operation .....	202
A.4.1	Information Transfer Handshaking .....	202
A.4.2	Address Transfer and Incrementing .....	204
A.4.3	Cache Immed-Invalidate Transaction .....	205
A.4.4	Cache Read-Shared Transaction .....	205
A.4.5	Cache Write-Back Transaction .....	205
A.4.6	Cache Read-Invalidate Transaction .....	206
A.4.7	Cache Write-Invalidate Transaction .....	206
A.4.8	Cache Read-Copy Transaction .....	206
A.4.9	Cache Write-Copy Transaction .....	206
A.4.10	Non-Cache Read Transaction .....	207
A.4.11	Non-Cache Write Transaction .....	207
A.4.12	Interlocked Transactions .....	207
A.4.13	Protocol Version .....	208
A.5	Timing Diagrams .....	208

## **Appendix B**

	<b>The Leopard-2 Bus Arbitration Protocol .....</b>	<b>222</b>
B.1	Arbiter and Protocol Description .....	222
B.2	VHDL Specification of the Arbitration Protocol .....	225

Appendix C  
A Behavioural Specification of Cache Coherence ..... 231  
References ..... 240

# Abstract

This thesis examines cache coherence protocols designed for use in bus connected shared memory multiprocessors. The cache coherency problem is discussed, and several previously published protocols are described in a new uniform framework, allowing ready comparison between them to be made. The issue of protocol correctness is also addressed. The protocol mechanisms proposed for the IEEE P896.2 Futurebus are described, and the way they may be used to implement the published protocols is illustrated. Two approaches for verifying correctness of the Futurebus mechanisms are described. A brief survey of techniques for evaluating relative performance of cache coherence protocols is presented, covering three main techniques: analytical, simulation based, and measurement of real systems. It is argued that the last of these is the most accurate, and that the results of such measurements are needed to validate evaluations based on the other two techniques. A brief description is presented of an early prototype multiprocessor, the Leopard-1, and a detailed description is presented of a full-scale multiprocessor system, Leopard-2. The Leopard-2 is an experimental platform which includes programmable cache controllers, designed to allow performance measurements of cache coherence protocols. Attention is focussed on the design of the cache attached to each processor, and the way in which the coherence protocols are implemented is described. A detailed behavioural model of the programmable cache controller is presented. The model is driven by two workloads: a synthetic workload to exercise specific aspects of system behaviour, and a pseudo-random workload to provide comprehensive test coverage. The model is used to verify correct maintenance of coherence by the caches operating under different coherence protocols. The thesis concludes with

a discussion of ways in which the experimental platform is used to **measure relative** performance of coherence protocols.