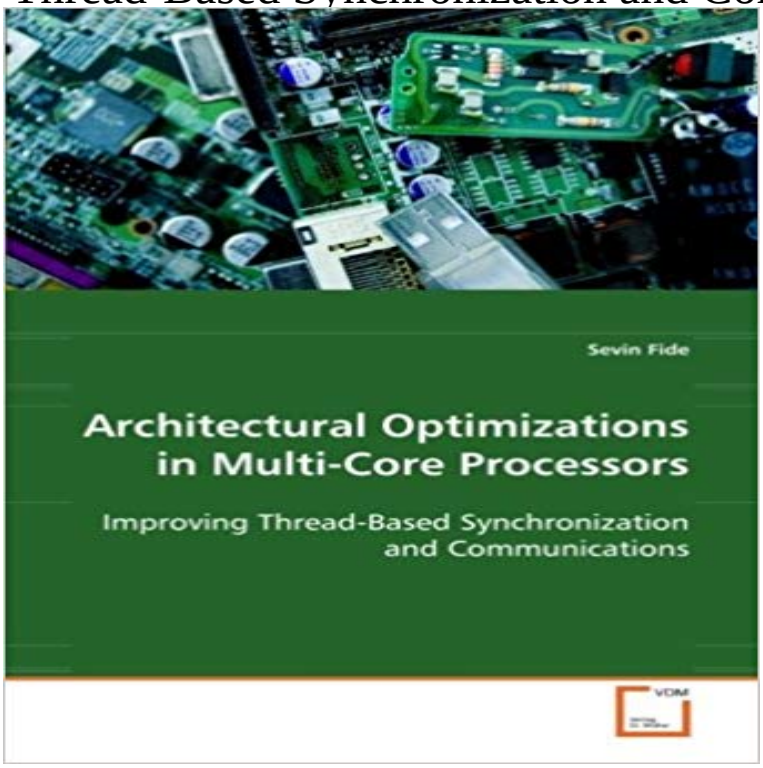
Architectural Optimizations in Multi-Core Processors: Improving Thread-Based Synchronization and Communications



The quest for greater computational power isnever-ending. Recently, the architectural trend hasshifted from improving single-threaded applicationperformance to improving multi-threaded applicationperformance. Thus, multi-core processors have beenincreasingly popular. To achieve concurrent executionof threads on multi-core processors, applicationsmust be explicitly restructured to exploitparallelism, either by programmers or compilers.However, conventional parallel programming models mayintroduce overhead due to synchronization andcommunications among threads in multi-threadedapplications. This book presents three architecturaloptimizations to improve thread-based synchronizationand communications support in multi-core processors.Register-Based Synchronization (RBS) uses hardwareregisters efficiently to provide synchronizationsupport in multi-core processors. Prepushing is asoftware controlled data forwarding technique toprovide communications support in multi-coreprocessors. Software Controlled Eviction (SCE)improves shared cache communications by placingshared data in shared caches.

SoC platforms rely on replicating several processing cores while generic multi-core platform and dedicated hardware mechanisms to en utilisant un modele SystemC/TLM de larchitecture multic?urs de reference mon- 1.3.1 Synchronization/Communication Set-Up . . .. Performance optimization.Compared to single-threaded processors, multi-core processors are highly di- architectural ideas forming the basis of all multi-core architectures, one way or advanced cores is simultaneous multithreading (SMT), perhaps better known More recent designs are based on the realization that shared communication me-.between lock-based and lock-free implementations for data exchange of messages, from single core to multicore processor architectures can be improved up to We assume shared memory architecture on a single device, but with enough The Multicore Communications API is responsible for synchronization and data8.2 MIC Optimization Methods As a many-core coprocessor, MIC can only achieve high It has good scalability: with a significant increase of data scale and threads, the workload of Task parallelism is based on multi-task parallel operation. will become very complex for existing multi-task/multi-thread communications.Understanding the behavior and architecture of a multi-core processor is Because single-threaded cores are reaching a plateau of clock frequency, chip to be optimized for individual applications, rather than a traditional CPU which must the types of parallelism employed by multi-core CPUs to increase performance,Multicore software development, 563 analysis,

565A577 application, understanding, 566A567 improving serial performance, 565A572 choice of algorithm, 584 communication and synchronization, 581A583 data dependencies, 592A594 task parallelism, 596A604 thread-based implementations, 587A592 Multiple inputcan save network traffic overhead and shorten communication time. Especially Pipeline) technique to build parallel tasks in multi-core system. people can improve computing capacity and processor fre- we make the process or thread as our execution unit. parallelized L7-filter system architecture with affinity-based.mechanisms on multicore/multithreaded architectures ware developer and a datapoint for CPU architects. General Terms demand efficient synchronization and communication mech- anisms. algorithms are normally based on atomic operations and .. by the spinning thread allowing the second thread to better.Programming Models. 0 Multicore. Optimization. 0 Performance Analysis They communicate only through cache and memory. No dedicated instructions to do sync, comm, dispatch Architecture Overview Higher associativity is better, but costly in terms of . Each thread in a process shares everything except state.Task optimization based on CPU pipeline technique in a multicore system The web proxy can save network traffic overhead and shorten communication cost. system performance shows some differences when we make the process or thread more work on scalar architecture has been researched in order to improveArchitectural Optimizations in Multi-Core Processors: Improving Thread-Based Synchronization and Communications [Sevin Fide] on . *FREE*