Shared Memory Consistency Models Evaluation in NoC based Multicore Systems

Abdul Naeem
Department of Electronic Systems, Royal Institute of Technology (KTH), Stockholm, Sweden
E-mail: abduln@kth.se

***Abstract*—This paper overviews our study on various shared memory consistency models, Sequential Consistency (SC), Weak Consistency (WC), Release Consistency (RC), and Protected Release Consistency (PRC) models in Network-on-Chip (NoC) based Distributed Shared Memory (DSM) multi-core systems. These memory models are implemented by using a *transaction counter (TC)* based *unified* approach in the NoC based systems. The performance gain observed in the WC, RC and PRC relaxed memory models under various benchmarks is between 20% and 50% compared to the SC strict model.**

I. INTRODUCTION

Network-on-Chip (NoC) is a scalable communication infrastructure for future multicore systems. Shared Memory can be Distributed (DSM) on-chip to exploit the high bandwidth of NoC. In DSM NoC based multicore (McNoC) systems, the shared memory transactions can be reordered due to several reasons and the systems may give unexpected results. Memory consistency imposes ordering constraints on the shared memory operations for the expected behavior of the DSM systems. Different memory consistency models (or memory models) enforce different ordering constraints on the shared memory operations [1]. This paper gives a summary of the study presented [2-6] on the shared memory consistency issue in the context of application specific McNoC systems. The following are the main contributions:

- Implementation of the *Shared Memory Consistency* models using a Transaction Counter (TC) based *unified approach*. The TC keeps track of the outstanding shared data (read, write) operations issued by a processor in each node of the network.
- Decoupled solution to the memory consistency issue, which is independent of cache/cache coherence protocol in the McNoC systems. While in general multiprocessor systems, both memory consistency and cache coherency are implemented using coherent caches which mainly suffer of scalability issue. We have studied the s*calability* of the memory models in the McNoC systems [2-6].
- The implementation schemes of the memory models are cheap and effective having small overhead (gates count) in the processor/network interface.
- Proposal of a new memory model, *Protected Release Consistency* (PRC) model as an extension of the well-known *Release Consistency* (RC) model [6].

The configurable McNoC platform is used for the experiments. The experimental results have shown [2-6] the performance gain of the relaxed memory models over strict memory models due to the additional relaxation in the shared memory operations. The hardware overheads are also compared.

In the following sections, various memory models, target architecture, implementations of memory models and performance evaluation of these models are briefly discussed.

II. SHARED MEMORY CONSISTENCY MODELS

A. Sequential Consistency (SC) Model

The SC model [4-5] is a strict memory model. According to SC model (Figure 1(a)), the shared memory operations are completed in the *program order* (order specified by the program). The *sequential order* is maintained by interleaving operations on lock S among processors in the system. The SC model does not allow reordering/relaxation among the shared memory operations. Hence, it cannot exploit the system optimizations [1] both in the hardware and in the software compared to the relaxed memory models. Some relaxed memory models are discussed next.

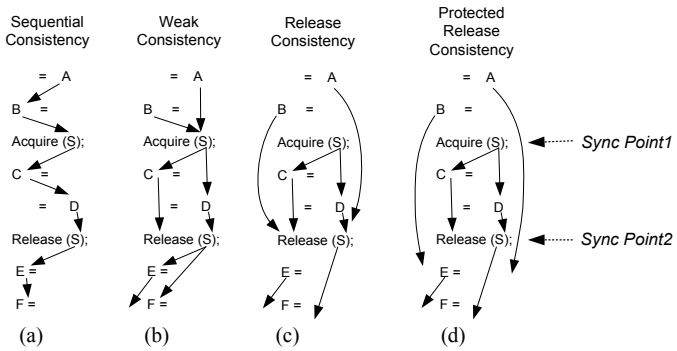

Figure 1. Comparison of SC, WC, RC and PRC models

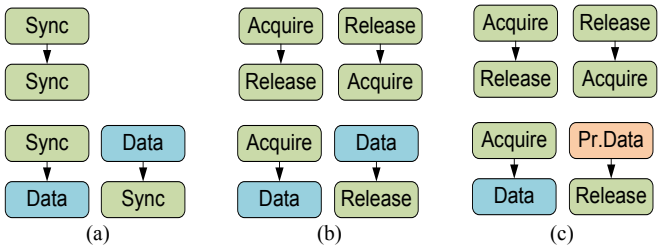

Figure 2. Global orders under a) WC model b) RC model c) PRC model
Sync: Synchronization, Pr.Data: Protected Data

B. Weak Consistency (WC) Model

The WC model [2-5] classifies shared memory operations as *synchronization (Sync)* and *data* operations. Sync

operations are related to the special Sync variables (locks, semaphores) in the shared address space. Data operations are the (read, write) operations related to the ordinary shared variables. The independent data operations issued between the two consecutive Sync points can be reordered with respect to each other as shown in Figure 1(b). The data operations are not allowed to be reordered with respect to the Sync operations and vice versa. The global orders to be enforced under the WC model are given in Figure 2(a).

C. Release Consistency (RC) Model

The RC model [2][5][6] is a refinement of the WC model. It further classifies the Sync operations as *acquire* and *release* operations. The independent data operations issued before a release operation or between two consecutive release points can be reordered up to the point of release operation. As illustrated in Figure 1(c), the data operations outside the acquire-release block are allowed to be moved into that block, but, the data operations in the acquire-release block are not allowed to be moved outside of that block. The global orders to be enforced under the RC model are given in Figure 2(b).

D. Protected Release Consistency (PRC) Model

The PRC model [6] further refines the RC model and classifies the data operations as *protected* and *unprotected* operations. Protected data operations are protected under acquire-release operations on a lock, others are unprotected data operations. As given in Figure 1(d), unprotected data operations issued *before* a release operation can be reordered with the release operation as well because they are independent of each other. The release operation *only* notifies the completion of the prior *protected* data and not that of the unprotected data operations. The global orders to be enforced under the PRC model are given in Figure 2(c).

III. Target Architecture

A configurable McNoC platform with typical processor-memory (PM) nodes is used in the tests [6]. The shared memory is distributed (*DSM*) in a global memory address space. The processor interface uses a *Transaction Counter (TC)* to implement memory models in the system. It also uses an output signal to control the flow of transactions from the processor. The platform uses the *distributed locks* in the global address space for the threads synchronization over critical shared memory references. The network interface performs packetization, de-packetization, queuing, and message passing and connects a PM node to the NoC. A *2D mesh packet-switched Nostrum NoC* [7] with deflection routing policy is used as a communication backbone.

IV. Implementation of Memory Models

The memory models are implemented by enforcing the required *global orders* (Figure 2) on the shared memory operations by using a TC based unified approach in the McNoC systems [2-6]. The SC model is implemented [4-5] by *stalling* the processor on the issuance of a shared memory operation till the completion of preceding operation. The processor issues the next operation on the completion of previously issued shared memory operation. The WC model is implemented [2-5] by using TC in each node of the network to keep tracks of the *outstanding data operations* issued between two consecutive Sync points. The TC is affected by the data operations and not by the Sync operations. The TC is checked at the Sync points to be zero. The RC model is implemented using *two* TCs approach [5] and *single* TC approach [6]. In [5], two TCs keep tracks of the outstanding data operations issued in critical and non-critical sections. In [6], TC keeps track of the outstanding data operations issued between two consecutive release operations. The TC is not affected by acquire and release operations. The TC is checked at the release points to be zero. In the *PRC model* [6], TC_P only keeps track of the *outstanding protected data operations* issued between the acquire-release block. The TC_P is not affected by acquire, release and *unprotected* operations. The TC_P is checked at the release points to be zero. In addition, *ordering constraints with respect to the lock operations* must also be enforced. For instance the operations on a lock must be completed in a sequential order. The lock must be gained before entering to the protected/critical section. The lock must not be released before the execution of protected section is completed.

V. Performance Evaluation

In the experiments [2-6, 8-9], the effects of network size, traffic patterns, different positions of the data/lock in the network and number of protected sections per node are investigated on the *code execution times* and *speedup* under memory models in the McNoC systems. The reduction in execution times observed under various *benchmarks* programs for the WC, RC and PRC models is between *20%* and *50%* compared to the SC model. The hardware overheads are also compared. The performance gain in the relaxed memory models depends on the *application programs*.

References

[1] S. V. Adve et al., "Shared Memory Consistency Models: A Tutorial," Digital Western Research Laboratory, report no. 95/7, USA, 1995.

[2] A. Naeem, X. Chen, Z. Lu, and A. Jantsch, "Scalability of Relaxed Consistency Models in NoC based Multicore Architectures," ACM SIGARCH Computer Architecture News, December 2009, 37(5): 8-15.

[3] A. Naeem et al., "Scalability of Weak Consistency in NoC based Multicore Architectures," in: Proceedings of IEEE International Symp. on Circuits And Systems (ISCAS), Paris, June 2010, pp. 3497-3500.

[4] A. Naeem et al., "Realization and Performance Comparison of Sequential and Weak Memory Consistency Models in Network-on-Chip based Multi-core Systems," in: Proc. of the 16th IEEE/ACM Asia and South Pacific Design Automation Conf. (ASP-DAC), 2011, pp. 154-159.

[5] A. Jantsch, X. Chen, A. Naeem, Y. Zhang, S. Penolazzi, and Z. Lu, Memory architecture and management in an NoC platform, in: Axel Jantsch and Dimitrios Soudris, editors, Scalable Multi-core Architectures: Design Methodologies and Tools, Springer, 2011.

[6] A. Naeem et al., "Realization and scalability of release and protected release consistency models in NoC based systems," in: Proc. of Euromicro Conf. on Digital Systems Design, DSD, 2011, pp. 47-54.

[7] A. Jantsch "The Nostrum NoC," in: http://www.ict.kth.se/nostrum.

[8] A. Naeem et al., "Architecture Support for Relaxed Memory Models in NoC based Distributed Shared Memory Systems," Submitted to: International Symposium on Network-on-Chip, (NOCS2012).

[9] A. Naeem et al., "Realization and Performance Comparison of Release and Protected Release Consistency Models in Adaptive NoC based Multi-core Systems," Submitted to: Microprocessors and Microsystems – (MICPRO) journal, Elsevier.