Chapter 3
Hardware Transactional Memories: A Survey

Arsalan Shahid
HITEC University, Pakistan

Muhammad Yasir Qadri
University of Essex, UK

Maryam Murad
HITEC University, Taxila Cantt, Pakistan

Nadia N. Qadri
COMSATS Institute of Information Technology, Pakistan

Jameel Ahmed
HITEC University, Pakistan

ABSTRACT

The initiation to have a concept of shared memory in processors has built an opportunity for thread level parallelism. In various applications, synchronization or ordering tools are utilized to have an access to shared data. Traditionally, multithreaded programming models usually suggest a set of low-level primitives, such as locks, to guarantee mutual exclusion. Possession of one or more locks protects access to shared data. But, due to some flaws they become a suboptimal solution. The idea of transactional memory is in research presently as an alternative to locks. Among which, one way is hardware transactional memory. Atomicity is well supported by using transactions in hardware. In this chapter, we have focused on hardware transactional memories and the work done on them so far.

INTRODUCTION

Two or more threads when needed to access some mutual data, then a proper mechanism is required to accurately make those executions. There were some issues in using locks to handle such flaws like mutual exclusion and debugging. Therefore, idea of transactional memory was presented (Guerraoui & Romano, 2014). Basic problem in programming multithreads is to manage shared states (Alessandrini, 2015). A great difficulty is faced when multithreaded programs are created because then the main problem that arises is to synchronize them and made them being able to access the mutual data properly

DOI: 10.4018/978-1-5225-0287-6.ch003

Copyright © 2016, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

(A McDonald, 2009). Programming and computer architecture communities are researching in finding ways to improve parallel techniques of accessing data (Navarro, Hitschfeld-Kahler, & Mateu, 2014). Database systems have successfully exploited parallel hardware for decades. Database is accessed by various operations through transaction even with predictable response.

The idea of transactional memory was first presented by Lomet (Lomet, 1985). He proposed an idea of atomic operations, like the one used in database systems, in programming languages. TM is an attractive feature that access parallel shared data by dealing intellectually with certain issues. TM system is based mainly on three major properties:

- **Atomicity:** States that operations within transactions are all finished successfully, or none of them is executed.
- **Consistency:** Means that each transaction initiates its operation with a consistent view of the shared data and exits the system in a consistent state after completion.
- **Isolations:** Concurrently running transactions are not interrupted by each other (Firoozshahian, 2009).

In TM, programmer defines certain regions that are executed at run-time to detect conflicts automatically. And this is done through read-set and write-set by tracking transactions, and checking reads and writes which are taking place within transactions. For atomicity and isolation, these read sets and write sets are compared with sets of other transactions (A McDonald, 2009).

Transaction Pattern

Transactional memory provides the following (see Table 1) special instructions to access memory instead of the simple load and store.

A transaction's read set is defined as the set of locations loaded using LT, write set as the set of locations loaded using LTX or stored using ST, and data set as the union of the two. The following (see Table 2) instructions are also provided to change the transaction state (Tripathi et al., n.d.).

Listing 1 shows a simple transaction style that shows how the above mentioned primitives can be implemented.

Listing 1. Transaction style

```
Repeat {
    Begin Transaction ();              /* initialize transaction */
    <Read input values>
    Success ← Validate ();             /* test if inputs consistent */
If (success)
    {
        <Generate updates>
        Success ← Commit ();           /* attempt permanent update */
        If (! success)
        Abort ();                      /*terminate if unable to commit */
    }
End Transaction ();                    /* close transaction */
}
Until (success);
```

Adapted from "Transactional Memory part 1".

17 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the publisher's webpage:
www.igi-global.com/chapter/hardware-transactional-memories/159039

Related Content

Multimedia Services Offer Mixing Telco and Internet Assets
Luis Angel Galindoand Joaquín Salvachúa (2012). *Grid and Cloud Computing: Concepts, Methodologies, Tools and Applications (pp. 1863-1884).*
www.irma-international.org/chapter/multimedia-services-offer-mixing-telco/64571

Service-Oriented Architectures for Pervasive Computing
Elias S. Manolakosand Demetris G. Galatopoullos (2010). *Handbook of Research on P2P and Grid Systems for Service-Oriented Computing: Models, Methodologies and Applications (pp. 147-174).*
www.irma-international.org/chapter/service-oriented-architectures-pervasive-computing/40801

A Workflow Scheduling Strategy for Reasoning Tasks of Autonomous Driving
Jianbin Liao, Rongbin Xu, Kai Lin, Bing Lin, Xinwei Chenand Hongliang Yu (2022). *International Journal of Grid and High Performance Computing (pp. 1-21).*
www.irma-international.org/article/a-workflow-scheduling-strategy-for-reasoning-tasks-of-autonomous-driving/304907

Advanced Topics GPU Programming and CUDA Architecture
Mainak Adhikariand Sukhendu Kar (2016). *Emerging Research Surrounding Power Consumption and Performance Issues in Utility Computing (pp. 175-203).*
www.irma-international.org/chapter/advanced-topics-gpu-programming-and-cuda-architecture/139843

Collaborative Edge Computing Advances in Children's Behavior Observation for Social Internet of Things Systems in Preschool Education
Danqing Liuand Luqun Liu (2022). *International Journal of Distributed Systems and Technologies (pp. 1-21).*
www.irma-international.org/article/collaborative-edge-computing-advances-in-childrens-behavior-observation-for-social-internet-of-things-systems-in-preschool-education/307952