Concurrent Dynamic Memory Coalescing on GoblinCore-64 Architecture

Xi Wang
Department of Computer Science
Whitacre College of Engineering
Texas Tech University
Box 43104
Lubbock, Texas 79409-3104
Email: xi.wang@ttu.edu

John D. Leidel
Department of Computer Science
Whitacre College of Engineering
Texas Tech University
Box 43104
Lubbock, Texas 79409-3104
Email: john.leidel@ttu.edu

Yong Chen
Department of Computer Science
Whitacre College of Engineering
Texas Tech University
Box 43104
Lubbock, Texas 79409-3104
Email: yong.chen@ttu.edu

***Abstract*—Mainstream modern microprocessor architectures are constructed with the memory systems that consist of multi-level data caches and traditional DDR main memory devices. The native hardware concurrency mechanisms present in the respective micro-architectural implementations only provide a low degree of hardware managed concurrency. Further, these mechanisms are often difficult or entirely not visible from the application layer or instruction set architecture. These mechanisms often promote efficient utilization or near-optimal performance for applications with significant memory reuse or linear memory access patterns.**

Conversely, applications generally considered to be *data-intensive* access memory in irregular and non-deterministic patterns or in strides that exceed the size of modern data caches. Executing this class of application on a traditional micro architecture has the inability to make use of the on-chip data caches, resulting in inefficient use of the memory hierarchy. In response to these data-intensive applications, we have developed the GoblinCore-64 (GC64) micro architecture with using a large degree of hardware-managed concurrency coupled to a high bandwidth memory subsystem [1].

We introduce the GC64 machine hierarchy in Figure 1. The core machine model and instruction set are based on the RISC-V [2] instruction set architecture. We utilize the three-dimensional stacked memory devices in the form of Hybrid Memory Cube (HMC) devices as the basis for the GC64 main memory. The HMC devices provide uniquely high bandwidth over traditional DDR-based memory units alongside a packetized memory interface. The GC64 system on chip consists of a series of hierarchical hardware modules. Each socket is constructed with one or more GC64 task groups. These task groups are integrated via a network on chip interface to four shared on-chip components. The on-chip software-managed scratchpad unit acts as a very high performance, user-mapped storage mechanism for commonly used data. The Atomic Memory Operation (AMO) Unit is responsible for controls queuing, ordering and arbitration of atomic memory operations. The HMC Channel Interface handles the protocol interaction between multiple HMC devices. Finally, the Off Chip Network Interface handles any off chip memory requests that utilize the GC64 memory addressing mechanisms.

Given that the GC64 micro architecture lacks inherent data caches as associated with the core memory hierarchy, the memory coalescing provided by the DMC unit is vital to optimizing the access to main memory from the concurrent task units. The HMC packetized request format provides a unique capability in that posting larger memory requests, up to 128 bytes each, significantly reduces the control overhead required to access main memory. As such, we construct a binary tree-based dynamic memory coalescing model to coalesce the memory requests from multiple, concurrent task units increases the efficiency of accessing main memory. Once the tree reaches the condition that triggers the expiration, the most left child will be found as the base address. We traverse the tree in order, checking each subsequent node in order to determine its spatial distance from the previous address. We do so recursively until we find the maximum possible request to inject into one or more HMC devices.

We present two parallel methodologies: Address Partitioned Algorithm (APA) and Work Partitioned Algorithm (WPA) as well as associated implementations for coalescing non-deterministic memory requests into the largest potential HMC request. In the APA, the tasks of dynamic memory coalescing are assigned to different and independent threads based on the address of memory requests. Each thread will only handle these addresses that fall within their assigned HMC address space partition and insert these requests into their local trees which are constructed following in the aforementioned coalescing tree logic. Based on partitioning the memory space, WPA also partitions the read and write operations. There will be two threads working on the same partition. The thread with a lower thread id (tid) will only handle the read requests. Correspondingly, thread with a higher tid will only insert the write requests that fall into the specified memory partition into its local coalescing tree [3]. In this manner, both APA and WPA increase the chance and efficiency of coalescing memory requests. We also present the coalesced HMC memory request results from applications that exhibit linear and non-linear memory request patterns compiled for a RISC-V core in contrast with a traditional memory hierarchy.

The contribution of this research study is three-fold:

- **Build the tree-based memory coalescing model with the tree logic design for correctly coalescing the memory accesses;**
- **Construct the architecture for concurrent DMC. Two different parallel algorithms are also designed for the concurrent DMC unit;**
- **Implement the concurrent DMC and prove the superiority of memory coalescing unit, in the perspective of efficacy through the test applications.**

REFERENCES

[1] John D. Leidel, Xi Wang, and Yong Chen. GoblinCore64: Architectural Specification. Technical report, Texas Tech University, September 2015.

[2] Andrew Waterman, Yunsup Lee, David Patterson, and Krste Asanovic. The RISC-V Instruction Set Manual, Volume I: User-Level ISA Version 2.0. Technical report, 2014.

[3] Xi Wang, John D. Leidel, Yong Chen. Concurrent Dynamic Memory Coalescing on GoblinCore-64 Architecture. MEMSYS 16 October 3-6, 2016, Washington, DC, USA. © 2016 ACM. ISBN 123-4567-24-567/08/06, DOI: 10.475/123_4.