Research on Cache Coherence Key Technology in Multi-core Processor System

su Zhang[1, a]
[1]Department of Computer Science，Gansu Normal University for Nationalities， Hezuo Gansu ,747000, China
[a]376788368@qq.com

Keywords: Cache, Coherence Key Technology, Multi-core Processor System

Abstract. The multi-core processor integrates more than one computing core into a single processor and enhances the processor's computing performance through parallel computing of multiple cores. Single-chip multi-processor architecture is the area of concern. This paper briefly discusses the CMP multi-level Cache storage structure, the multi-level structure of the cache caused the consistency problem and the choice of consistency agreement has an important impact on CMP system performance. What kind of Cache consistency model is used and its design scheme are the main contents of this paper.

Introduction

In the past two decades, the computer processor design process and processor architecture has developed rapidly, the computer can also complete most of the tasks assigned to it. At the same time, a wide range of applications has also brought strong demand for high-performance and low-power processors. In the past by integrating more transistors in a single-core processor to enhance the processor frequency method, with the Moore's Law and the processor power dissymmetry asymmetry, making the high-frequency processor will bring the previously unpredictable work Consumption problem.

Multi-core processor is to integrate multiple computing cores in a single processor, through its parallel computing technology to improve the computing power of the processor. Therefore, for high-performance multi-core processor research, microprocessor research has become an important part of the field. In the past, multi-core technology is mainly used in mainframes, servers and workstations and other high-end products, with multi-core processors in general desktop and mobile devices (high-performance notebook) in the popularity of high performance and low power multi-core microprocessors The application of the device will be further to mobile phones, PDAs and other mobile electronic products in the expansion. It is in this trend, the study of multi-core microprocessors will be more extensive and in-depth, its scope of application will be further expanded. With the rapid multi-core processor market, multi-core processor will certainly be one of the hot spots of the future.

Multi-core processor in the typical structure - single-chip multi-processor architecture is the field of research. CMP refers to the integration of multiple microprocessor cores on a single chip. Each microprocessor core is essentially a relatively simple single-threaded microprocessor, and multiple microprocessor cores can execute program code in parallel. Intel announced at the 2005 Beijing Summit ("Intel Information Technology Summit" in Beijing) that the focus of research and development in the next 5-10 years will shift to CMP structures. This shows that, CMP structure will be the future development trend of microprocessor structure. In the microprocessor structure continues to develop at the same time, storage technology is also evolving. Although the speed of the memory is increasing, but its speed is far less than the microprocessor's processor frequency, there is still a large speed gap between them.

The Common Causes of the Problem in the Consistency

There are three reasons for inconsistency: 1. Inconsistencies caused by sharing of writable data; 2.

Inconsistencies caused by process migration; 3. Inconsistencies caused by bypassing Cache's I / O operations. Each of the possible cases of inconsistency is discussed below.

Inconsistencies Caused by Sharing of Writable Data. If the processor P1 writes a new data Y 'to the cache, if the write pass policy is used to immediately update the main memory, in this case, two copies of the two copies Y and Y' are inconsistent; Writeback strategy, only when the modified data in the Cache is replaced or become invalid, the main memory of the content was updated, it will also produce inconsistencies.

The Inconsistencies Caused by Process Migration. If the write-through policy is used, processor P1 modifies the variable to Y 'and updates the value of the variable in main memory to Y'. When the process containing the shared variable transitions from processor P1 to processor P2, the variable value in P2 remains Y , If the write-back policy is used, the process changes the value of the variable in the private Cache of the processor P1 to Y '. When the process moves from the processor P1 to the processor P2, because of the write-back policy, P2 will get from the main memory variable value, is still Y, there is inconsistency.

The Inconsistencies Caused by I/O Operations Bypassing Cache. When the I/ O processor writes a new data Y 'to main memory, bypassing the cache with the write-through policy produces inconsistencies between C1 and main memory. When bypassing the Cache directly from the main memory output data, the use of write-back strategy Cache will produce inconsistencies. For example, DMA data transfer, DMA controller directly to the memory operation (read or write), but each time there may be a corresponding copy of the Cache data, it will cause inconsistencies between memory and Cache.

Cache for inconsistencies caused by the process of migration can be prohibited by the process of migration to solve the process can also be suspended at the time by the hardware method of the Cache in the process of rewriting the information block forced to write back to the main memory corresponding location to solve; For I / O operations caused by inconsistencies, a direct method is to I / O processor and each dedicated cache directly connected to form the main processor shared Cache structure, I / O operation can guarantee consistency.

The Analysis of Existing Protocols

Monitoring Protocols. The listening protocol uses a shared bus to connect the private and main memory of multiple processor cores. The shared bus ensures that data requests from all the processor cores are executed serially. Data requests from any processor are broadcast to all processor core nodes, and the cache controller of all processor cores monitors the bus at all times. If a read request is received, the data owner returns the valid data to the processor core that issued the request; if a write request is received, the processor core with the valid copy is invalid or updates the data on the processor, the data owner valid data will be returned to the processor core that issued the request. The data owner may be the processor core, or it may be main memory.

The snoop-based protocol broadcasts consistent messages to all nodes on the chip to request the required data. The listener can quickly see the request and respond immediately, so the cache miss latency is small in small-scale systems. With the increasing size of multi-core, monitoring protocol will face serious challenges: First, the multi-core data requests in the release of the shared bus depends on the complete sequence of features, each time there can only be a processor core, multiple communication components on the bus of the contention request will cause the phenomenon of blocking access, latency is too large lead to poor scalability is a listening protocol in the public nuclear system, the most important limiting factor; Second, the listening protocol in the obsolete, update operation will be carried out The system-wide message broadcast, on-chip interconnection structure must be a long load of great traffic, on-chip components need to monitor the bus dynamic, the endpoint of the degree of activity will increase, resulting in on-chip power issues cannot be ignored.

Directory-Based Protocols. The directory protocol uses a directory to hold valid copies of data in the private cache of all processor cores, ensuring that all processor cores access the same data serially via the directory. Any data request from the processor core will be sent to the directory first,

the processor kernel node containing the valid replica of this data will be searched through the directory, and then the data request will be sent to these nodes. The processor core with valid replica will send the valid data Returns to the private cache of the processor core that issued the request, or invalidates or updates the data on the processor. This is the so-called "triple jump" process or "Cache-to-Cache Missing" process.

The initial directory protocol system maintains the shared state information for all addresses, called the "full-mappings" directory protocol. Later, the researchers in order to reduce the storage space occupied by the global directory, the directory protocol made a number of optimization and improvement techniques, including limited directory, chain directory. The limited directory protocol limits the size of each directory entry to reduce the amount of storage space the directory occupies. If the directory storage space is full, some of the processor's valid copies are written back to the shared storage to free up directory pointer storage to satisfy the new Claim. The chained directory protocol organizes the directories in the form of linked lists, and the shared storage and private cache store the directory pointer information.

Token Protocol. Token conformance breaks the traditional protocol design idea, and considers the correctness and performance of cache coherence in the design. It constructs the underlying framework to ensure correctness, avoids the request of hunger while guaranteeing the safe operation of the protocol, and can be based on the correctness framework builds multiple performance strategies.

The correctness framework guarantees security through the Token counting mechanism: the system sets a fixed number of Tokens for each data block in a shared storage system, and when a processor is to read a block, it must have a Token for that block; when writing to the block, it must have all the tokens for that block. This Token counting mechanism directly guarantees that there is only one write operator or multiple read operators at any one time, and there is no data inconsistency. Second, the correctness framework uses the persistence request mechanism to avoid hunger that occurs when certain requests cannot be satisfied for a long time. Once the persistent request is activated, the system will always forward the Token until the initiator of the persistent request finally obtains the required data and the corresponding read and write permissions. Finally, the performance strategy focuses on higher data throughput and bandwidth utilization, through the transient request mechanism (Transient Requests) to the system components and To-ken data request.

Hammer Protocol. AMD Hammer protocol used in multi-machine system can actually be applied to multi-core processor, it is the listener protocol and directory mechanism combined to expand, so that it can adapt to the disordered structure of the inter-connect. When a processor requires cache access, the request is sent to the Home block of the requesting block. The Home node acts as a role in sorting access requests for the same data block, making the protocol applicable to disordered on-chip interconnect structures. Home node does not rely on the directory mechanism for data response or forwarding request, but directly to the access request to broadcast in the form of forwarding to other nodes in the system. The Home node responds to the requester with data from local storage, and the other node sends an acknowledgment to the requester that the obfuscation of the corresponding data block has been successfully completed.

Compared with the listening protocol, the Hammer protocol uses a similar broadcast mechanism in the forwarding request, but there is no order requirement for the interconnection structure, and it needs to gather the acknowledgment information of each node. Compared with the directory protocol, Hammer does not need special Directory structure to preserve the corresponding data block status information, but rely on the Home node to complete the directory sorting point function.

The Hot Spots and Challenges of Multi-Core Cache Consistency Mechanism

The on-chip interconnect architecture and the Cache coherence protocol are tightly coupled in the design implementation. First, the bandwidth provided by the interconnection structure has an impact on the scalability of the Cache coherence protocol. Second, the cache coherence protocol may

require a sequential interconnection structure to ensure correctness (eg, listening protocol). The low latency, high bandwidth, scalable interconnect architecture is the hardware foundation for implementing the efficient Cache Coherence protocol. Designers want to simplify the consistency-preserving mechanism by using sequential interconnects, such as buses, and to provide highly scalable low-latency communication frameworks with disordered interconnect structures such as on-chip networks, The high bandwidth required to provide a sequenced interconnect structure and provide a large scale system is often not available.

Multi-core processors require frequent fine-grained data interactions between components, and a large number of request commands, data responses, and acknowledgment messages are generated when maintaining a cache coherency mechanism, Interconnection structure of the traffic and the active node, the system brings heavy power consumption burden. In a cache coherency protocol, it is often necessary to send the same consistency message (eg obsolete) to multiple storage nodes rather than to all components, so if the multicast mechanism is used to send the consistency request only to the necessary subset of nodes , The power consumption due to the broadcast communication can be reduced. It is an effective means to reduce the power consumption by searching multicast mechanism suitable for specific interconnection structure instead of system-wide broadcast message.

The Cache Coherence Protocol is a key factor in ensuring that shared storage systems behave correctly. Designers often need to invest a lot of effort in verifying the correctness of the protocol to ensure that it satisfies the given constraints, such as model checking and other formalization Methods, and even simulation-based design validation. Cache protocol verification process includes checking data consistency, protocol incompleteness, whether there is a live lock and deadlock. Formal verification at the beginning of the design is very helpful in finding defects in the conformance agreement.

The Cache Consistency Protocol fault tolerance mechanism and correctness are two different concepts. Cache consistency protocol is responsible for ensuring that the memory access request of the processor can be processed in the correct order, so that multiple processors check the shared memory space has a consistent view; fault tolerance mechanism refers to the consistency of the agreement on the sudden abnormal situation Such as the loss of messages caused by transient error injection in the on-chip interconnect structure, and the like. IC process technology features continue to shrink, increasing frequency, making the interconnection structure, including on-chip components, including transient error more sensitive.

Conclusion

With the development of multi-core processors, CMP with the characteristics such as its control logic is simple, high frequency, low communication delay and its application prospects are very good. However, due to the use of multi-level cache hierarchical structure, it causes a serious Cache inconsistency. In this paper, the existing protocols are analyzed and studied, the advantages and disadvantages of the two protocols are summarized and a new consistency protocol is designed to improve the performance of the system.

References

[1] Huifang Zhou: High Performance Computing Technology, Vol. 6 (2014) No 53, p.25-26

[2] Hongli Zhang: Technology Frontiers, Vol. 12 (2015) No 27, p.74-76

[3] Qin Guo: Application of Electronic Technique, Vol. 1 (2011) No 33, p.11-14

[4] Jieming Liu: Computer Engineering and Applications, Vol. 3 (2012) No33, p.121-124

[5] Lin Wang: Computer information, Vol. 1 (2011) No 33, p.21-25