



White Paper  
**Liang-min Wang, Ph.D**  
Platform Applications  
Engineer  
Intel Corporation

# How to Implement a 64B PCIe\* Burst Transfer on Intel® Architecture

February 2013



## ***Executive Summary***

---

To transfer data between two memory regions through PCIe\* transactions, there are two alternatives:

- Through DMA configuration
- Through the core bus interface unit

In this paper, we focus on the second method and describe a way to transfer 64B of data in a single burst through the CPU core bus interface unit (BIU) to a memory region (outside of the CPU core and not part of cache hierarchy) reserved for a PCIe device. The fundamental principle of this approach is to change the cache attribute of the PCIe device memory region. With this approach, the IO property of PCIe memory is closely retained (no cache is used for retaining data on core) and a long burst PCIe transfer is feasible. This method makes the best use of system bus and PCIe bus bandwidth.

The Intel® Embedded Design Center provides qualified developers with web-based access to technical resources. Access Intel Confidential design materials, step-by step guidance, application reference solutions, training, Intel's tool loaner program, and connect with an e-help desk and the embedded community. Design Fast. Design Smart. Get started today. [http://www.intel.com/p/en\\_US/embedded](http://www.intel.com/p/en_US/embedded).



## Contents

---

Problem .....	4
Solution .....	4
How to Mark a Memory Region as WC.....	5
How to Ensure Read/Write Ordering when Accessing a Memory Region Marked as WC .....	6
Conclusion .....	7



## Problem

---

A memory mapped IO register or memory region is marked as un-cacheable by the system default setting. The system default setting could be changed from the BIOS or OS kernel routines. Each reference to these resources from the program is executed immediately and the maximum burst size for data transferred in and out of the CPU core is limited by the CPU load/store bus width. On the other end, saving IO bounded resources on the core cache reduces cache bandwidth and provides no performance benefit. This paper describes a data transfer method that keeps PCIe\* memory data out of cache and uses the wider BIU bus width (64 byte).

## Solution

---

IA processor system memory can be partitioned into various regions, and each region can be assigned a unique cache attribute. A brief description of three of these cache attributes is provided below:

- WB (WriteBack)
- UC (UnCacheable)
- WC (WriteCombine)

See the *Intel64 and IA-32 Architectures Software Developer's Manual* [1] for a full description of cache attributes.

System memory where data are accessed without side-effect<sup>1</sup> is marked as WB so the system can take advantage of high-speed data access from various cache entries available from system cores. The data marked as WB attribute keeps a copy of data in a core cache entry (depending on the cache fetch policy and cache hierarchy, the same data may optionally occupy one entry per cache level) until the respective cache-line is evicted from the core. The trade-off of this attribute is that memory data coherence between cache and memory device is not "retained" all the time until either a memory fence instruction is issued or a respective cache line is evicted by hardware events such as cache line replacement.

Data marked as UC and WC are assigned to a memory region that keeps no copy in cache entries, and reads and writes to such a memory region delivers data directly from/to the target memory device. The difference between these

---

<sup>1</sup> A memory read/write side-effect can be described as either a read or write side effect. A read side-effect doesn't guarantee that two successive reads to the same memory region will return the same value. A write side-effect doesn't guarantee that a read followed by a write to the same memory location returns the same value that was written the same location.



two attributes is that a memory region marked as WC enables the core write-buffer controller to combine successive write requests to maximize the core BIU bus bandwidth. The implication is that a write request to a WC memory region doesn't guarantee that the write data is immediately delivered to the external memory device and coherence is not guaranteed. A read or write to a UC memory region delivers data from/to the BIU immediately. In addition to the delayed write, there is no speculative read allowed on memory regions marked as UC.

The maximum data size that can be delivered by each read/write from/to memory marked as UC depends on the internal load/store buffer bus width. For second generation or later Intel® Core Processors this bus is 32B wide.

To enable a 64B burst write, this paper presents an approach that modifies Memory Type Range Register (MTRR) through programming Model Specific Registers (MSR) to create a memory region marked as WC and issues memory fence instructions that ensure each 64B is delivered once the data is available in spite of how the write-buffer combine logic is designed.

## How to Mark a Memory Region as WC

Intel architecture provides a set of MSRs to change default system behavior such as cache attributes, performance counters, etc. Reading and writing to any MSR is done using RDMSR and WRMSR instructions. These are privileged instructions; accessing MSRs is only available in system mode (ring 0).

A set of MTRRs [1] are available for marking the cache attributes of a memory region. There are two types of memory regions that can be marked with a specified cache attribute. One is a fixed-region MTRR setting and the other is a variable-region MTRR setting. Besides the per-region setting, additional MTRRs are available to describe default behavior (IA32\_MTRR\_DEF\_TYPE) and system MTRR capability (IA32\_MTRRCAP) [1]. Since programming the MTRR may change read/write behavior of the target memory region, the *Intel64 and IA-32 Architectures Software Developer's Manual* [1] provides rules that must be followed when writing to MTRRs. For fixed-region MTRR programming, the memory address is limited between 0 and 1MB.

**Note:** For Linux\*, kernel functions are available for accessing MTRRs, please refer to `asm/mtrr.h` for the detailed API definition.

To program a variable-region MTRR, two MTRRs must be programmed: base register and mask register. The mask register provides the range information. When a fixed-region MTRR conflicts with a variable-region, the fixed-region specification takes precedence.

**Caution:** Caution needs to be taken on accessing data crossing two regions that are programmed with different cache attributes. A memory access transaction that crosses between UC and WB regions may produce unexpected behavior.



## How to Ensure Read/Write Ordering when Accessing a Memory Region Marked as WC

The IA architecture supports instructions to ensure memory access ordering. There are three instructions designed to support memory access serialization:

- LFENCE: Load Fence instruction that guarantees serialization of pending load instructions
- SFENCE: Store Fence instruction that guarantees serialization of pending store instructions
- MFENCE: Fence instruction that guarantees serialization of all pending memory load/store instructions

Once the desired PCIe memory region is marked as WC, a burst transfer of 16B is done through:

```
_mm128_store_si128(pcie_memory_address, xmm0);  
_mm_mfence();
```

The memory fence instruction is required so that the preceding store instruction won't be delayed for possible write-combine. To run a 64B burst transfer, the code below is recommended:

```
_mm256_store_si256(pcie_memory_address, ymm0);  
_mm256_store_si256(pcie_memory_address+32, ymm1);  
_mm_mfence();
```

To avoid a cross-cacheline penalty for a 64B burst transfer, the `pcie_memory_address` must be 64B aligned. For a 16B transfer the address must be 16B aligned.



## Conclusion

---

This paper presents a way to enable a 64B burst transfer from IA processors to PCIe end-point devices using cache attributes and fence instructions. Since 64B is the bus width of the core BIU, the same technique can be applied for wider burst transfers if future processors support wider BIU bus width.

**Note:** The integrated PCIe controller interface may provide additional capability to support burst combine to increase burst length. For more information, refer to the programming guide for the target PCIe control interface.

The Intel® Embedded Design Center provides qualified developers with web-based access to technical resources. Access Intel Confidential design materials, step-by step guidance, application reference solutions, training, Intel's tool loaner program, and connect with an e-help desk and the embedded community. Design Fast. Design Smart. Get started today. [http://www.intel.com/p/en\\_US/embedded](http://www.intel.com/p/en_US/embedded).

### Authors

**Liang-min Wang** is a Platform Application Engineer in the Intelligent Systems Group at Intel Corporation.

### Acronyms

BIU Bus Interface Unit: This is the bus interface between the core and un-core.

MTRR Memory Type Range Register: Registers used to mark cache attributes for a memory region

MSR Model Specific Register: Registers used to modify the default behavior of the processor

UC UnCacheable: cache attribute that does not keep a copy of data in a core cache entry

WB WriteBack: cache attribute that keeps a copy of data in a core cache entry

WC WriteCombine: cache attribute that does not keep a copy of data in a core cache entry but may be delayed and combined with other writes

### Reference

[1] Chapter 11 Memory Cache Control of Intel64 and IA-32 Architectures Software Developer's Manual Volume 3: System Programming Guide, May 2012



INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to:  
<http://www.intel.com/design/literature.htm%20>

Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

\*Other names and brands may be claimed as the property of others.

Copyright © 2013 Intel Corporation. All rights reserved.