

PowerPC™

Application Note **Multiprocessor Systems and the PowerPC 603e™ Microprocessor**

This application note describes some of the issues confronting the systems designer when the 603e or PowerPC 603™ processors are used to implement a multiprocessor system. Although the 603e (and the 603, unless otherwise noted) does not provide hardware support for multiprocessor system functions provided by the PowerPC 604™ processor, many of the hardware mechanisms of the 604 that allow efficient operation in multiprocessing systems can be provided by operating system software routines. This document describes attributes of the 603e that require operating system software support to facilitate multiprocessor system operation.

Additional information about the topics discussed in this document can be found in *PowerPC 603e RISC Microprocessor User's Manual* (order #: MPC603EUM/AD), and *Addendum to PowerPC 603e RISC Microprocessor User's Manual: PowerPC 603e Microprocessor Supplement and User's Manual Errata* (order #: MPC603EUMAD/AD).

In this document, the terms '603', '603e', and '604' are used as abbreviations for 'PowerPC 603 microprocessor', 'PowerPC 603e microprocessor', and 'PowerPC 604 microprocessor', respectively. These microprocessors are available from Motorola as MPC603, MPC603e, and MPC604, respectively.

Note that the 603e is implemented in both a 2.5-volt version (PID 0007v PowerPC 603e microprocessor, abbreviated as PID7v-603e) and a 3.3-volt version (PID 0006 PowerPC 603e microprocessor, abbreviated as PID6-603e). The scope of this document encompasses both implementations of the 603e unless otherwise noted.

The PowerPC name, the PowerPC logotype, PowerPC 603, PowerPC 603e, and PowerPC 604 are trademarks of International Business Machines Corporation used by Motorola under license from International Business Machines Corporation.

This document contains information on a new product under development by Motorola. Motorola reserves the right to change or discontinue this product without notice.

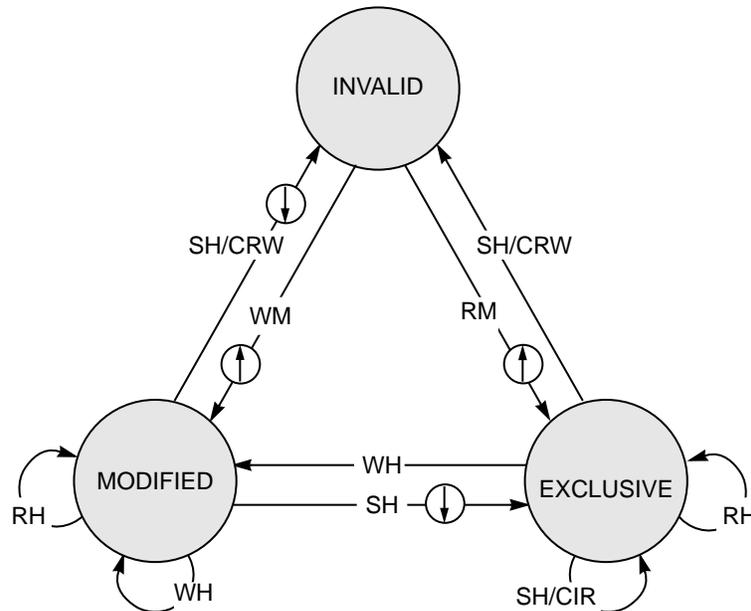
© Motorola Inc. 1996. All rights reserved.



To locate updates for this document, refer to the website at <http://www.mot.com/powerpc/>.

1.1 MEI Cache Coherency Protocol

The 603e cache coherency protocol is a subset of the modified, exclusive, shared, and invalid (MESI) four-state cache protocol, the 603e implements only MEI, as shown in Figure 1. Cache block loads from pages marked globally coherent are identified during bus transactions as write misses (read-with-intent-to-modify operations), causing other 603e processors to flush (if the snooped cache block is modified) or invalidate (if the snooped cache block is unmodified) the corresponding copies of data from their cache. When a 603e has completed a cache block load, it is the exclusive owner of the data and may write to it without performing an address-only bus transaction indicating the cache state change from exclusive to modified.



Bus Transactions

| | |
|-----------------|--|
| SH = Snoop Hit | SH/CRW = Snoop Hit, Cacheable Read/Write |
| RH = Read Hit | SH/CIR = Snoop Hit, Caching-Inhibited Read |
| RM = Read Miss | ⬇ = Snoop Push |
| WH = Write Hit | ⬆ = Cache Line Fill |
| WM = Write Miss | |

Figure 1. MEI Cache Coherency Protocol—State Diagram (WIM = 001)

Figure 1 shows the operation of the MEI protocol for memory pages or blocks with WIM settings of 001 (write-back, caching-not-inhibited, and memory coherency enforced). One exception to the treatment of a read operation by the 603e as a snooped write is when the snooped operation is a caching-inhibited read (either single-beat or burst). If the 603e gets the snoop hit on a modified cache block, it writes the block back to memory and marks the block exclusive unmodified. If the cache block is marked exclusive unmodified when the caching-inhibited read operation is snooped, no bus action is taken and the cache block remains in the exclusive unmodified state. This treatment of caching-inhibited reads reduces the possibility of data thrashing by letting noncaching devices read data from the 603e's data cache without invalidating the entry.

Systems with multiple 603e processors sharing the bus can reduce the number of bus transactions caused by read-induced cache block invalidations and flushes by minimizing the number of pages and blocks with memory coherency enforced (WIM = 001).

The sharing of program instructions among multiple 603e processors is not subject to the same coherency restrictions enforced for data, because no snooping is performed in the 603e instruction cache.

1.2 TLB Synchronization

The 603e provides translation lookaside buffer (TLB) structures to maintain on-chip copies of the page table entries (PTEs) resident in memory. Following power-up, hard reset, or in the course of certain operating system operations, the TLB entries must be explicitly cleared before valid entries are loaded. The Translation Lookaside Buffer Invalidate Entry (**tlbie**) instruction is provided for the invalidation of TLB entries. Execution of the **tlbie** instruction by the 603e does not cause an address-only bus transaction to signal other 603e processors to invalidate their matching copies of the PTE, and the 603e does not invalidate TLBs in response to a TLB invalidation broadcast by any other processor. Synchronization of TLB entries in a multiple 603e system must be performed by the operating system or other software.

However, the 603e provides a $\overline{\text{TLBISYNC}}$ signal input to allow for hardware synchronization of changes to TLBs and PTEs when multiple 603e processors or direct-memory access (DMA) devices share the same MMU translation tables. While the $\overline{\text{TLBISYNC}}$ signal is asserted, the 603e halts instruction execution after executing a Translation Lookaside Buffer Synchronization (**tlbsync**) instruction until the $\overline{\text{TLBISYNC}}$ signal is negated. If the **tlbsync** instruction is executed with the $\overline{\text{TLBISYNC}}$ signal negated no internal or external action occurs. With the use of external control logic, the $\overline{\text{TLBISYNC}}$ signal can provide a locking and updating mechanism for the TLBs and PTEs of 603e processors in a multiprocessor system.

The Translation Lookaside Buffer Invalidate All (**tlbia**) instruction is not implemented on the 603e, and when its opcode is encountered, an illegal instruction program exception is generated. To invalidate all entries in both TLBs, 32 **tlbie** instructions must be executed, incrementing the value in EA[15–19] by one each time. The **tlbie** instruction will invalidate four TLB entries when executed, clearing both the instruction and data translation lookaside buffer (ITLB and DTLB) entries indexed by EA[15–19].

1.3 Cache Management Instructions and Bus Broadcasting

Table 1 provides an overview of the bus operations initiated by cache control instructions. None of the instructions broadcast address-only transactions on the bus except the Data Cache Block Set to Zero (**dcbz**) instruction, and no broadcasts by other bus masters are snooped by the 603e, with the exception of cache block kill transactions. When the HID0[ABE] bit is set the PID7v-603e broadcasts address-only transactions when the Data Cache Block Invalidate (**dcbi**), Data Cache Block Flush (**dcbf**), and Data Cache Block Store (**dcbst**) instructions are executed. The 603e coherency logic does not provide a snoop response to the **dcbi**, **dcbf**, and **dcbst** instructions; therefore, the system designer must ensure that the correct response is provided.

Table 1. Bus Operations Caused by Cache Control Instructions (WIM = 001)

| Instruction | Cache State | Next Cache State | Bus Operation | Comment |
|--------------|--------------------|------------------|---------------------|--|
| sync | Don't care | No change | None | Waits for memory queues to complete bus activity |
| icbi | Don't care | Invalid | None | — |
| dcbi | Don't care | Invalid | None (Kill block*) | — |
| dcbf | Invalid, exclusive | Invalid | None (Flush block*) | — |
| | Modified | Invalid | Write-with-kill | Block is pushed |
| dcbst | Invalid, exclusive | No change | None (Clean block*) | — |
| | Modified | Exclusive | Write | Block is pushed |

Table 1. Bus Operations Caused by Cache Control Instructions (WIM = 001) (Continued)

| Instruction | Cache State | Next Cache State | Bus Operation | Comment |
|---------------|---------------------|------------------|----------------------------|---|
| dcbz | Invalid | Modified | Write-with-kill | — |
| | Exclusive, modified | Modified | Kill block | Writes over modified data |
| dcbt | Invalid | No change | Read | Fetches cache block is stored in touch load queue |
| | Exclusive, modified | No change | None | — |
| dcbtst | I | No change | Read-with-intent-to-modify | Fetches cache block is stored in touch load queue |
| | Exclusive, modified | No change | None | — |

Note: Bus operation performed when PID7v-603e HID0[ABE] is set.

1.4 Memory Synchronization Using **lwarx** and **stwcx**. Instructions

Memory synchronization instructions control the order in which memory operations are completed with respect to asynchronous events and the order in which memory operations are seen by other processors or memory access mechanisms. The Load Word and Reserve Indexed (**lwarx**) and Store Word Conditional Indexed (**stwcx**.) instructions allow programmers to emulate common semaphore operations such as ‘test and set,’ ‘compare and swap,’ ‘exchange memory,’ and ‘fetch and add.’ Examples of these operations can be found in Appendix E, “Synchronization Programming Examples,” in *PowerPC™ Microprocessor Family: The Programming Environments* user’s manual. Both the **lwarx** instruction and its corresponding **stwcx**. instruction must have the same effective address. Note that the reservation granularity is 32 bytes.

The concept behind the use of the **lwarx** and **stwcx**. instructions is that a processor may load a semaphore from memory, compute a result based on the value of the semaphore, conditionally store it back to the same location (only if that location has not been modified since it was first read), and determine if the store was successful. If the reservation exists when the store is executed, the store is performed and the CR0[EQ] bit is set. If the reservation does not exist when the store is executed, the target memory location is not modified and the CR0[EQ] bit is cleared.

If the store is successful, the sequence of instructions from the read of the semaphore to the write that updated the semaphore appear to execute atomically (that is, nothing modified the semaphore location between the read and the update), thus providing the equivalent of a real atomic operation. However, in reality, other processors may have read from the location during this operation. A read operation that hits a reserved cache block invalidates the cache block but does not clear the reservation. The subsequent execution of an **stwcx**. instruction to the invalidated cache block holding the reservation causes a single-beat write to memory, after which the reservation is cleared. A single-beat write operation is also performed if the cache block addressed by the **stwcx**. instruction is valid (that is, the **stwcx**. instruction is treated as a write-through operation).

While only one reservation can exist on any processor, more than one processor in a multiprocessor system can concurrently reserve a cache block. The address associated with the reservation can be changed by a subsequent **lwarx** instruction. The conditional store is performed based upon the existence of a reservation established by the preceding **lwarx** regardless of whether the addresses match. A reservation held by the processor is cleared either by executing an **stwcx**. instruction to any address or by an attempt by some other device to modify a location in the reservation granularity (32 bytes).

Information in this document is provided solely to enable system and software implementers to use PowerPC microprocessors. There are no express or implied copyright licenses granted hereunder to design or fabricate PowerPC integrated circuits or integrated circuits based on the information in this document.

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters can and do vary in different applications. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part.

Motorola and  are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

IBM is a trademark of International Business Machines Corporation.

The PowerPC name, the PowerPC logotype, PowerPC 603, PowerPC 603e, and PowerPC 604 are trademarks of International Business Machines Corporation used by Motorola under license from International Business Machines Corporation. International Business Machines Corporation is an Equal Opportunity/Affirmative Action Employer.

Motorola Literature Distribution Centers:

USA/EUROPE: Motorola Literature Distribution; P.O. Box 20912; Phoenix, Arizona 85036; Tel.: 1-800-441-2447

JAPAN: Nippon Motorola Ltd.; Tatsumi-SPD-JLDC, Toshikatsu Otsuki,
6F Seibu-Butsuryu-Center, 3-14-2 Tatsumi Koto-Ku, Tokyo 135, Japan; Tel.: 03-3521-8315

HONG KONG: Motorola Semiconductors H.K. Ltd.; 8B Tai Ping Industrial Park,
51 Ting Kok Road, Tai Po, N.T., Hong Kong; Tel.: 852-26629298

MFAX: RMFAX0@email.sps.mot.com; TOUCHTONE (602) 244-6609

INTERNET: <http://Design-NET.com>

Technical Information: Motorola Inc. Semiconductor Products Sector Customer Support Center; (800) 521-6274.

Document Comments: FAX (512) 891-2638, Attn: RISC Applications Engineering.

World Wide Web Address: <http://www.mot.com/powerpc/>

