基于P6总线的多处理器系统Cache一致性设计*)

张江陵　刘劲松　冯　丹

(信息存储系统教育部重点实验室　华中科技大学　武汉430074)

摘　要　本文介绍了基于P6总线的多处理器系统的总线事务和存储区的Cache属性,讨论了P6总线的硬件监听机制,Pentium Ⅲ处理器所采用的MESI状态转换,最后研究了多处理器和P6总线如何相互配合以保证整个系统的Cache一致性。

关键词　监听,Cache一致性,MESI协议,P6总线,多处理器系统

Cache Coherency Design in Multiprocessor System Based on P6 Bus

ZHANG Jiang-Ling　LIU Jin-Song　FENG Dan

(Key Laboratory of Data Storage System, Huazhong University of Science and Technology, Ministry of Education, Wuhan 430074)

Abstract　After introducing bus transaction and cache attribute of memory area in multiprocessor system based on P6 bus, this paper discusses hardware snoopy mechanism of P6 bus, then discusses MESI state transitions adopted by Pentium III processor, and finally focuses on how the multiprocessor and the P6 bus cooperate to ensure Cache coherency of the whole system.

Keywords　Snoop, Cache coherency, MESI protocol, P6 bus, Multiprocessor system

1　引言

总线型多处理器系统一般采用基于监听的MESI协议来保证系统的Cache一致性。由于体系结构的不同以及对性能和硬件设计方面的考虑,不同的总线型多处理器系统中硬件监听机制的具体实现以及MESI状态转换也不相同。本文研究基于P6总线的多处理器系统Cache一致性问题。P6总线可以支持Intel的多种处理器,如Pentium Pro/Pentium II/Pentium III,构成多处理器系统。本文只讨论Pentium III,其它处理器情况类似。

Intel公司没有公开Pentium III处理器Cache设计的所有细节。在其发布的技术资料中[1,2,5~7],也未将多处理器系统的Cache一致性作为一个专门的技术论题给出其全貌,而是在讨论一些其它相关论题的时候有所涉及,因而非常零碎和不完整。文[3]对Intel公司多处理器系统Cache一致性问题有所探讨,但较局限于处理器内部的Cache结构和工作方式。

尽管Cache只是处理器内部的一个组成部件,但Cache一致性设计却是一个全局性的问题。它不仅涉及处理器内部Cache的结构,还涉及系统中存储区域的分布、总线层次结构、访存总线事务等多方面因素,因而必须从系统全局的角度来研究。

本文首先讨论了P6总线多处理器系统的概况,包括系统存储区域的分布、访存总线事务、存储区域Cache属性控制等。其次分析了P6总线的监听机制和处理器两级Cache的MESI状态转换。最后从系统全局的角度研究了多个处理器的各级Cache和P6总线如何相互配合以实现整个系统的Cache一致性。

2　系统概况

2.1　存储区域分布及访存分析

P6总线多处理器系统基本结构如图1所示。系统被划分为三个层次:P6总线层、PCI总线层、X总线层(ISA/EISA/二级PCI等)。由HOSI/PCI桥和PCI/X桥将它们连接为一个整体。

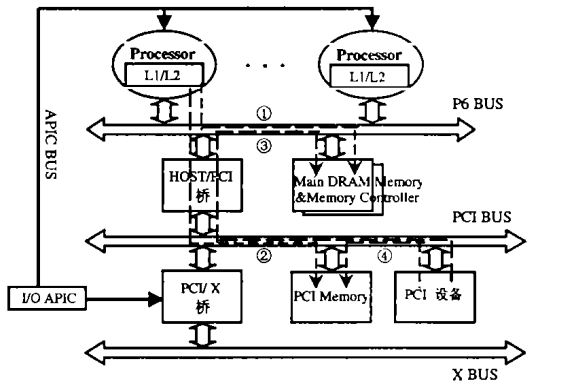


图1　P6总线多处理器系统基本结构

系统中,Pentium Ⅲ处理器支持的4GB物理内存空间(不考虑36位扩展)大致分为两类:一类是主存区,占据从物理地址0开始的低端地址空间,上限不超过HOST/PCI桥的TOM值(Top Of Memory);另一类是I/O设备内存区,范围为TOM~4GB-1[6]。

针对上述内存区的访问,根据访问的发起者也可分为两类:一类是由处理器发起的访问;另一类是由I/O设备发起的访问,例如由PCI Master设备或由ISA DMA设备发起的访问。根据访问的发起者和访问目标划分,存在四类不同的访存活动,它们涉及不同的总线事务。图1中,处理器对主存的访问将启动P6总线事务;处理器对PCI Memory的访问将首先启动P6总线事务,然后由HOST/PCI桥将其映射为PCI总

*)本文由全国优秀博士学位论文专项基金资助.刘劲松　博士研究生,主要研究方向为并行存储系统.张江陵　教授,博士生导师,主要研究方向为计算机存储体系结构.冯　丹　教授,主要研究方向为附网储存系统.

线事务;PCI Master 对 PCI Memory 的访问将启动 PCI 总线事务;而 PCI Master 对主存的访问将首先启动 PCI 总线事务,然后由 HOST/PCI 桥将其映射为 P6总线事务。

P6总线多处理器系统中,每个处理器拥有 L1/L2两级 Cache,同一数据在内存和各 Cache 中可能有多份拷贝。如不加以控制,上述各种访存活动都有可能引起数据在各拷贝之间的不一致。基于总线的硬件监听技术正是用于解决这一问题的,它利用对总线上访存事务的监听而获知各种访存信息,进而采取相应的措施。这里就涉及这样一个问题,即监听到哪一级总线,硬件实现的代价如何。上述各种访存活动涉及不同总线层次的事务,因此若要用单纯的硬件监听手段来保证整个系统的一致性,就必须对系统中所有的总线进行监听,这很困难。由于历史的原因,ISA/EISA 总线根本就没有提供监听手段;PCI 总线虽然提供了有效的监听手段[4],但硬件实现开销较大。

在实际的 P6总线多处理器系统中,只实现了 P6总线级的硬件监听。这种方案用硬件机制来保证对主存访问的 Cache 一致性,硬件开销小,但带来的一个后果是,不能保证对 I/O 设备内存访问的 Cache 一致性,必须由软件机制加以补全。

2.2 存储区域 Cache 属性控制

Intel 提供了有限程度的软件机制来控制 P6总线多处理器系统中各存储区的 Cache 属性[1,3]:通过 BIOS 对处理器 MTRRs 寄存器组编程可实现区域级 Cache 属性控制;通过操作系统对页表 PTE 的 PCD/PWT 控制位编程可实现粒度为4kB 的页级 Cache 属性控制。通过软件编程,可以将各存储区的 Cache 属性设为下列5种之一:UC(uncacheable),WC(write combining),WP(write protected),WB(write back),WT(write through)。

对于 I/O 设备内存,系统初始化时由 BIOS 将其 Cache 属性设为 UC 或 WC,即对该区的访问不做 Cache 映射。这一方面的原因是对其监听的硬件实现开销较大,另一方面更重要的原因是,映射到 I/O 设备内存区的数据通常是诸如 PCI 设备的控制寄存器组、状态寄存器组、视频帧缓冲数据等,处理器对这类数据的访问没有多少局部性可言[1],对其做 Cache 映射不会带来多少性能方面的好处。处理器对 I/O 设备内存(UC/WC)的访问,其数据不进入 Cache,而是直接对内存单元进行操作,因而 I/O 内存中总是保存着最新和唯一的数据拷贝,也即保证了该区数据在整个系统中的一致性。

对于主存,系统初始化时通过 BIOS 和操作系统来控制主存 Cache 属性,使其不同存储单元呈现不同的 Cache 属性。处理器的 Cache 策略受主存 Cache 属性的影响,针对具有不同 Cache 属性的主存行的访问,其映射的不同 Cache 行策略(Cache 行与行之间)也不一样。一般情况下,WB(写回)属性的存储单元可以获得最佳的访存性能,这也是 BIOS 和操作系统对主存中大部分区域 Cache 属性的通常设置。

P6总线多处理器系统中,通过 BIOS 和操作系统软件设置各内存区域的 Cache 属性,从而在系统4GB 的物理内存空间中确定了硬件机制所负责的区域和相应的 Cache 策略。这种软硬件结合的方法减小了系统的硬件开销,又不影响系统的访存性能。本文下面将重点分析系统硬件机制如何保证对主存中 WB 类型存储区访问时的 Cache 一致性。

3 P6总线及 L1/L2 Cache[7,8]

基于 P6总线的多处理器系统中,P6总线负责连接处理器、HOST/PCI 桥、主存及主存控制器等主要部件,构成监听的硬件基础。P6总线上的部件称为代理(Agent),分为三类:请求代理、响应代理、监听代理。请求代理是总线事务的发起者;响应代理是事务的目标;监听代理是事务中除请求代理外的所有具有 Cache 的设备,它负责监听 P6总线上的访存活动。

P6总线操作以事务的形式出现。事务一般有6个相位,与 Cache 一致性处理相关的是请求、监听、响应和数据相位。

·请求相位中,请求代理通过 A[35:3]# 和 REQ[4:0]# 在两个时钟周期中提交本次事务的地址信息、存储区域 Cache 属性、数据长度和请求方案,并被总线上其他代理锁存。这些代理利用锁存的信息可判断出两点:一是确定谁是本次事务的响应代理,二是监听代理用来判断是否监听命中。

·监听相位中,监听代理通过 HIT#/HITM# 将监听结果报告给其他代理。HIT# 和 HITM# 信号组合值为:00仍需等待监听结果;01干净行命中;10脏行命中;11监听未命中。

·响应相位中,根据事务的请求方案和采样的监听结果,由响应代理通过 RS[2:0]# 将本次事务的响应方案提交给其他代理。

·数据相位中,根据响应方案完成相应的数据传输。

Pentium III 处理器有 L1/L2两级 Cache,双独立总线结构。L1通过总线接口部件 BIU 与外部 P6总线相连,L2通过专用内部 Cache 总线与 BIU 相连。

L1分指令 Cache 和数据 Cache。对于指令 Cache,除自修改代码(Self-modifying Code)和交叉修改代码(Cross-modifying Code)等特殊情况外[1],L1指令 Cache 不会发生写命中和写监听命中,它只实现了 MESI 缓存状态的 S/I 子集,状态转换较简单,本文在随后的部分就只讨论 L1数据 Cache。L1数据 Cache 位于处理器核心与主存间 look through 位置。对于 WB 型主存行,写命中策略为 write back,写丢失策略为 write allocate,且为先填入 Cache 行后修改。L1 Cache 控制器既可以通过 BIU 在 P6总线上启动访存事务,又负责监听总线上其他代理的访存事务。

L2是 L1的大容量备份。L2 Cache 控制器有两个监听窗口,内监听窗口负责监听本处理器 L1 Cache 的写命中活动,外监听窗口负责监听 P6总线上其他代理的访存事务。

4 P6总线多处理器系统 Cache 一致性的实现

P6总线多处理器系统采用基于监听的写无效 MESI 协议来维护 Cache 一致性。当采用写回策略时(对应主存行 Cache 属性为 WB),Pentium Ⅲ处理器 L1/L2中的相应 Cache 行均实现了 MESI 全部四个缓存状态,定义如下:M(Modified)为修改态,此行已被修改过(脏行),为最新数据且为此 Cache 专有,主存中为过时数据;E(Exclusive)为专有态,此行内容同于主存(干净行)且一定不出现于其他处理器中;S(Shared)为共享态,此行内容同于主存(干净行)且可能出现于或不出现于其他处理器中;I(Invalid)为无效态,此行内容已作废。

4.1 L1 Cache 的 MESI 状态转换

Pentium Ⅲ处理器 L1的 MESI 状态转换如图2(a)所示。

当 Pentium Ⅲ处理器作为一次访问的发起者启动对一个数据的读/写操作时,L1位于处理器核心与主存之间的 look through 位置,工作过程如下:

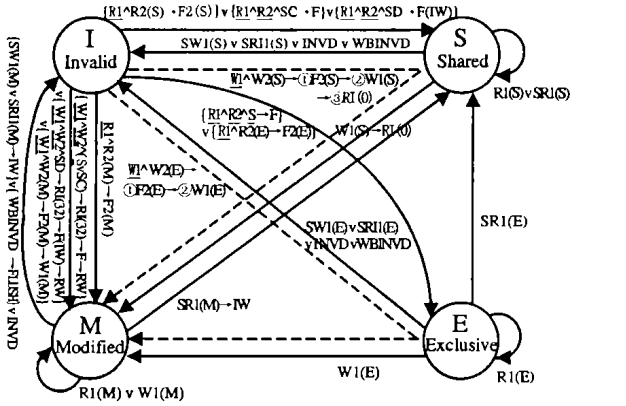
1)首先在 L1中进行查找,若命中,则处理器核心直接对 L1进行读/写操作,此时除 S 态写命中的情况外,读/写操作及状态变迁均在该处理器内部完成,不产生片外总线事务,因而该操作对总线上其他代理透明。在 S 态写命中的情况下,

Cache 行变为 M 态，并在 P6总线上发起一个0字节读并使无效总线事务，使该行数据在其他处理器上可能有的 S 态拷贝作废；

2)若 L1读/写丢失，则通过处理器内部 Cache 总线启动查询，看该处理器的 L2是否命中。若 L2读/写命中，L2有效行连同其状态经内部专用 Cache 总线拷贝到 L1中，然后在 L1中完成该行的读/写命中处理，包括相应的状态转换；

3)若 L2读/写丢失，则该处理器作为请求代理在 P6总线上启动一个访存总线事务。此时分为两种情况：①若 L1/L2为读丢失，则总线事务为32字节存储器数据读，读取的整行数据同时拷贝到 L1/L2中，其状态均为 S 态或 E 态，取决于此时其他处理器 Cache 有无监听命中。从数据源的角度来看，若监听结果为监听丢失或干净行(S/E)监听命中，则从主存中读入该行数据；若监听结果为脏行(M)监听命中，则由监听代理通过隐式写回将脏行数据同时写回主存和作为请求代理的处理器的两级 Cache 中。②若 L1/L2为写丢失，则总线事务为32字节存储器读并使无效，读取的整行数据只拷贝到 L1中，然后在 L1中进行写修改操作，新行最终状态为 M 态，并作废此行在其他处理器 Cache 中可能有的拷贝。从数据源的角度来看，若监听结果为监听丢失或干净行(S/E)监听命中，则从主存中读入该行数据到作为请求代理的处理器 L1中；若监听结果为脏行(M)监听命中，则由监听代理通过隐式写回将脏行数据写回作为请求代理的处理器 L1中。此时主存是否接收监听代理隐式写回的脏行数据取决于主存控制器的设计。由于 Pentium Ⅱ处理器 L1写丢失策略为先读入后修改的 write allocate，在此情况下主存中仍是过时数据，因而主存控制器一般设计为忽略32字节读并使无效总线事务中的隐式写回，以节省主存带宽。

当 Pentium Ⅱ处理器作为总线事务的监听代理时，若为读监听命中，L1行状态变为 S 态；若为写监听命中或读并使无效监听命中，L1行状态变为 I 态。若监听命中时 Cache 行状态为 M，则除了自身的状态变迁外，通常还有一个脏行数据隐式写回的过程。从时间顺序上看，状态变迁先于隐式写回；在总线事务的错误相位结束后，监听代理内部 Cache 就可以进行自身的状态变迁，此时将脏行数据拷贝到处理器内部的一个32字节写回 buffer 中，直到数据相位时，才发生隐式写回操作，从写回 buffer 而非 Cache 中获取隐式写回的数据。



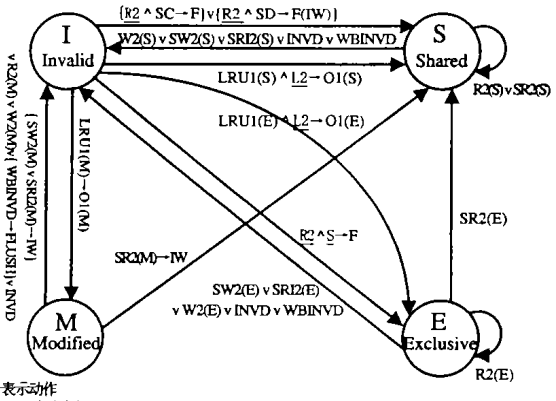
注：1. 下述符号表示事件
①当本处理器作为请求代理时：
Rx(y)：在 Lx 级 cache 中状态为 y 的 cache 行发生读命中
Wx(y)：在 Lx 级 cache 中状态为 y 的 cache 行发生写命中
Rx：在 Lx 级 cache 读丢失
Wx：在 Lx 级 cache 写丢失

②当本处理器作为监听代理时：
SRx(y)：在 Lx 级 cache 中状态为 y 的 cache 行发生读监听命中
SWx(y)：在 Lx 级 cache 中状态为 y 的 cache 行发生写监听命中
SRIx(y)：在 Lx 级 cache 中状态为 y 的 cache 行发生读并使无效监听命中

③当本处理器作为请求代理时系统中其他监听代理发生：
S：监听丢失
SC：干净行(E/S)监听命中
SD：脏行(M)监听命中

④其它事件：
LRUx(y)：在 Lx 级 cache 中状态为 y 的 cache 行发生替换操作
L2：该行在 L2 cache 中不存在

图2(a)　Pentium Ⅱ处理器 L1数据 Cache MESI 状态转换



2. 下述符号表示动作
RI(n)：本处理器作为请求代理在 P6 总线上发 n 字节读并使无效请求
IW：本处理器作为监听代理发起32字节隐式写回
F(IW)：本处理器作为请求代理接收由监听代理 M 态行32字节隐式写回
F：由主存读入一行
F2(y)：将本处理器 L2 cache 中的状态为 y 的 cache 行连同其状态一起拷入 L1 cache 中
RW：先读入一行到 L1 cache 中然后修改
Ox(y)：将 Lx 级 cache 中状态为 y 的 cache 行置换出
FLUSH：将 cache 中所有脏行数据全部写回上存

3. 下述指令将影响 cache 状态：
INVD：使 cache 行无效（without write back）
WBINVD：使 cache 行无效（with write back）

4. 其他
v 逻辑或
^ 逻辑与
→ 直接转换
⇢ 间接转换

图2(b)　Pentium Ⅱ处理器 L2 Cache MESI 状态转换

4.2　L2 Cache 的 MESI 状态转换

Pentium Ⅱ处理器 L2的 MESI 状态转换如图2(b)所示。

L2读/写命中和读/读并使无效/写监听命中的定义及其相应动作与 L1有所不同。①L2读命中指，L1读丢失时查找 L2命中，此时将命中的 L2行连同其状态(M/E/S)经内部专用 Cache 总线拷贝到 L1中，L2中的原 S 态或 E 态行保留，原 M 态行拷贝后变为 I 态。②L2写命中指，L1写丢失时查找 L2命中，此时 L2并不发生写修改操作，而是将 L2行连同其状态拷贝到 L1中，然后 L2行变为 I 态，拷入 L1的有效行则根据拷入的状态按写命中情况进行相应的状态转换。③L2读监听命中(读并使无效监听命中)指，P6总线上其他请求代理启动存储器读(读并使无效)总线事务时，L2通过外监听窗口进行监听，若命中则叫读监听命中(读并使无效监听命中)，此时 L2有效行变为 S 态(I 态)。④L2写监听命中指，当 P6总线上其他请求代理启动存储器写总线事务时，L2通过外监听窗口监听；或本处理器核心对 L1写命中时，L2通过内监听窗口监听，上述两种情况下的命中都称为 L2写监听命中，此时 L2有效行变为 I 态。

4.3　替换算法

当 L1/L2行发生替换操作时，一组4行中按 LRU 算法选择一行换出。若换出行是 L1的 M 态行(简记为 M1，下同)，则并不替换到主存中，而是将该行连同其状态一起经内部 Cache 总线拷贝到 L2中。若换出行是 M2行，则将该行数据拷贝回主存。若换出行是 S2或 E2行，则简单作废即可。若换出行是 S1或 E1行，则有两种可能的操作方法：方法一是简单作废；方法二是启动对 L2的查询，若该行在 L2中有备份，则简单作废即可，若该行在 L2中没有备份，则将 S1/E1行连同状态拷贝到 L2中。Intel 公司未公布 Pentium Ⅱ处理器的 LRU 替换算法的细节，上述两种针对 S1/E1替换的操作方法都是可行的，方法一实现起来较简单，方法二在 Cache 命中率方面有一定的好处，但替换操作代价略大一些。图2(b)表示的是方法二。

图2(a)(b)描述了 Pentium Ⅱ处理器 L1/L2在各种情况下的状态转换及相应动作，即"在什么事件发生的情况下→产生怎样的动作→进行怎样的状态转换"。例如图2(a)中 I1→S1状态转换的逻辑表达式为"{R1^ R2(S)→F2(S)}v{R1^

R2^ SC→F}v{R1^ R2^ SD→F(IW)}",其含义是有三种情况导致L1行由I态变为S态(状态转换):①在L1读丢失且L2的S态行读命中时(情况),将S2行连同状态拷入L1(动作);或者②在L1/L2均为读丢失且监听结果为干净行监听命中时,由主存读入一行;或者③在L1/L2均为读丢失且监听结果为脏行监听命中时,由监听代理将M态行隐式写回。

4.4 I/O设备访问主存时的Cache一致性

除处理器访问主存外,还必须保证I/O设备访问主存时的Cache一致性。HOST/PCI桥是所有I/O设备在P6总线上的代理。I/O设备提出访问主存的请求时,首先启动一个PCI总线事务,然后由HOST/PCI桥将其映射到P6总线上,启动一个或多个P6总线事务。此时,桥作请求代理,多个处理器作监听代理,主存及主存控制器作响应代理。桥作请求代理与处理器作请求代理并无本质的不同,但在总线事务为存储器写访问且监听结果为脏行监听命中时,情况有点特殊:①如果PCI总线命令是Memory Write,则映射到P6总线上的事务是一个8字节(或其子集)存储器写,由于监听结果为脏行监听命中,因而响应方案为隐式写回,主存控制器将隐式写回的脏行数据与PCI数据(已写到桥芯片post buffer中了)合并,再写入主存。②如果PCI总线命令是Memory Write and Invalidate,则映射到P6总线上的是一个或多个32字节存储器写事务,此时即使P6总线监听结果为脏行监听命中,但由于是整行写,因而响应方案为无数据方案,不发生脏行数据的隐式写回,主存控制器直接接收PCI数据(在桥芯片post buffer中),并写入主存。

4.5 综合

下面讨论P6总线多处理器系统中各处理器的L1/L2 Cache与P6总线如何相互配合实现Cache一致性。

假设某一时刻系统中数据单元Ω所在行的初始状态为:处理器A处的Cache行状态为I1-I2,处理器B处的Cache行状态为M1-I2,主存中为过时数据。又设处理器A对Ω写操作。

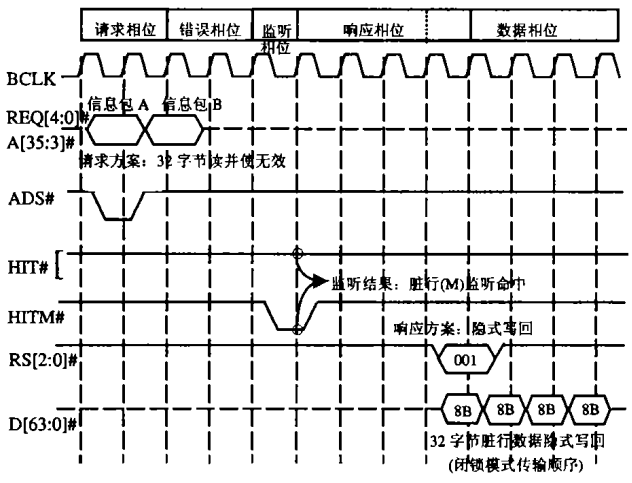


图3 32字节读并使无效总线事务(脏行监听命中)时的简化时序图

处理器A首先在它的L1中查找,结果为写丢失,于是通过专用Cache总线对它的L2启动查询。由于L2也为写丢失,于是通过总线接口部件BIU在P6总线上启动一个32字节读并使无效总线事务,该总线事务的时序如图3所示:①在该事务的请求相位中,处理器A作为请求代理,通过A[35:3]#提交本次事务的地址信息,通过REQ[4:0]#指明本次事务类型为32字节读并使无效。P6总线上其他代理锁存这些信息,进而判断出本次事务的响应代理是主存及主存控制器,而监听代理(处理器B)则判断出自己的L1发生了M态行读并使无效监听命中。②在该事务的监听相位中,监听代理(处理器B)一方面在片外HIT#/HITM#信号引脚上发出'10'信号组合,表示脏行监听命中,供其他代理采样;另一方面在其内部按图2(a)进行状态转换M1→I1,并将该行数据复制到处理器B的一个内部32字节写回buffer中。③在该事务的响应相位中,主存控制器根据在监听相位中采样的HIT#/HITM#监听结果,提出本次事务的响应方案为隐式写回,并通过RS[2:0]#信号线将响应方案提交给处理器A/B。④在该事务的数据相位中,监听代理(处理器B)负责完成32字节脏行数据的隐式写回,处理器B捕获数据总线控制权,将写回buffer中的数据加载到D[63:0]#上,每个总线时钟周期传递8个字节,共用至少4个时钟周期按总线闭锁模式传输顺序(取决于主存交叉存储结构和访存的起始地址),将写回bufefr中的数据写入到请求代理(处理器A)中。在此过程中,响应代理(主存控制器)忽略隐式写回的数据,即主存数据不做任何更新。从处理器A的角度来看,此行数据只拷贝到L1中,然后在L1中进行写修改操作,其最终状态为M态。

在此过程中,处理器A的状态转换为I1→M1,转换条件及动作为W1^ W2^ SD→RI(32)→F(IW)→RW。处理器B的状态转换为M1→I1,转换条件及动作为SRI1(M)→IW。该总线事务结束后,整个系统关于数据单元Ω的最终状态为:处理器A处Cache行状态为M1-I2,处理器B处Cache行状态为I1-I2,主存中依然为过时数据。

结束语 尽管Cache只是处理器内部的一个组成部件,但一致性设计却是一个全局性的问题。在讨论存储区域分布、访存总线事务、存储区域Cache属性控制等系统概况的基础上,分析了P6总线的监听机制和处理器两级Cache的MESI状态转换,最后从系统全局的角度研究了多个处理器的各级Cache和P6总线如何相互配合实现整个系统的Cache一致性。

参考文献

1 IA-32 Intel Architecture Software Developer's Manual, Volume 3: System Programming Guide. Intel Corporation. 2002

2 http://developer.intel.com/design/processor/

3 Shanley T. Pentium Pro and Pentium II System Architecture, 2nd Edition. Mindshare, Inc. 2001

4 Shanley T, Anderson D. PCI System Architecture, 3rd Edition. Mindshare, Inc. 1995

5 Developer's Insight CD-ROM, Intel Corporation. 1998

6 Intel 440BX AGPset: 82443BX Host Bridge/Controller Datasheet. Intel Corporation. 1998

7 Pentium Pro family developer's manual, volume 1: specification. Intel corporation, Nov. 1997

8 张昆藏. 奔腾II/III处理器系统结构. 电子工业出版社, 2000