序言

- 大家好，下面由我们组进行小组展示，我们组展示的主题是缓存一致性。

引入

- 首先由我来为大家介绍缓存一致性问题，以及一种缓存一致性协议。
- 那么什么是缓存一致性问题呢？
    - 我们来看下面的例子
        - 两个处理器读取了同一个变量的值，这个变量被加载到了他们的缓存中。
        - 然后 P0 修改了变量的值
        - 这时，P1 中这个变量的值已经过时了，但是 P1 的缓存中这个变量还是有效的
        - 同一个数据在 P0 和 P1 缓存中的值不一致，这就是缓存一致性问题。
- 那么我们如何解决缓存一致性问题呢？
    - 在上面的例子中，如果 P0 在修改的时候能把新值传播给 P1，那么这个问题就解决了
    - 因此我们需要一个机制来将一个缓存中的修改传播到其他缓存中，这种需求被称为写传播。
    - 写传播有两个主要的策略，写无效，写更新。
    - 写无效就是在写入的时候无效化其它的缓存，当其它处理器读的时候会重新加载最新的值
    - 写更新就是在写入的时候更新其它的缓存

- 下面我来介绍广播侦听式的 MESI 缓存一致性协议
    - 对于广播侦听式协议，
        - 每个缓存都有一个一致性控制器，它们向总线广播请求，或是从总线侦听请求
        - 在需要读写数据时，缓存控制器会尝试获取总线的权限，在总线上进行广播
        - 这些处理器通过总线上广播和侦听与其它处理器或内存通信，因此叫做广播侦听式一致性协议
    - 同时对于每个缓存行，在 MESI 协议中有四种可能的状态
        - M独占可写、E独占只读、S共享只读、I无效
        - 对应 MESI 的四个字母
    - 下面我们举例来看一下这个协议是如何工作的
        - 首先P0发出读请求
            - 由于其它缓存中都没有对应的数据
            - 因此内存响应P0的读请求，把数据读出来放在总线上
            - P0从总线上取回数据，标记为E独占状态
        - 然后P0发出写请求
            - 由于缓存块的状态为E独占状态
            - 因此一致性控制器不用在总线广播
            - 直接把缓存块状态标记为M独占状态，开始写入
        - 然后P1发出读请求
            - 由于P0中已经有了这个数据，P0会响应P1的读请求
            - P0将缓存块标记为S共享状态，发出传输请求并把数据放在总线上
            - P1从总线上取回数据，并标记为S状态
        - 然后P1发出写请求
            - 由于缓存块的状态为S共享状态，因此P1的控制器会广播无效化请求
            - P0监听到无效化请求，将缓存块标记为无效状态
            - 随后P1标记缓存块为M独占状态，开始写入
    - 在整个过程中，通过广播和侦听，MESI协议保证了系统的缓存一致性
    - 但是在大规模的多处理器系统中，广播和侦听的开销是非常大的，下面由ghy同学为大家详细讲解这个问题