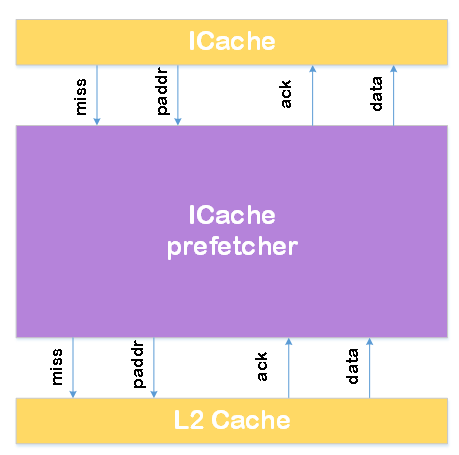
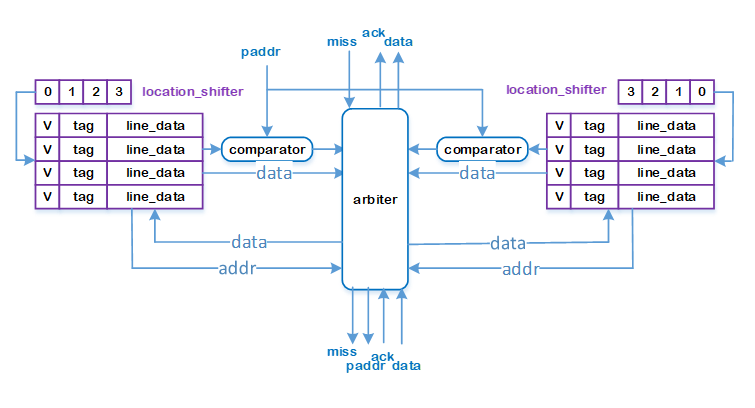
# ICache prefetcher

ICache prefetcher是ICache和L2 cache中间的一个数据便利店，核心是一个stream buffer结构。它截获ICache的miss信息，检查自己buffer中的存储信息是否可以满足ICache的需求，如果满足，直接将数据提供给ICache，ICache的miss交易完成。如果不存在ICache请求的数据，则由prefetcher向更大一级数据仓库L2 Cache发起数据请求，并将数据反馈给ICache。同时Prefetcher会根据ICache的请求信息来提前准备一些未来ICache可能的需求数据。ICache prefetcher和Icache和L2 Cache之间的接口如下图所示，加入prefetcher后ICache和L2 Cache的接口和原来ICache和L2 Cache的接口保持一致。



下图是ICache prefetcher的内部结构示意图。Prefetcher由一个两路的stream buffer组成，每路4项，为FIFO结构。选择两路是因为指令执行活跃的指令流只有一个，但是因为存在可能的分支预测错误，2路指令流可以让正确和错误的指令流同时存在，在分支纠错后的取指效率会提高。



工作原理：

当发生ICache miss的时候，miss地址会率先查找stream buffer，左右两路buffer的第一个元素会被比较。我们可以使用一个包含四个buffer地址的location shifter随着buffer中数据的流动进行移位，这样shifter的第一个地址总是指向buffer中的第一个元素。如果miss地址在任何一路buffer中命中，则只需要一个时钟周期就可以将数据从stream buffer中搬移到ICache中，即一个时钟周期就完成ICache miss的处理。数据被搬出buffer后，对应的valid位会置0，同时location shifter的地址会移动一次。

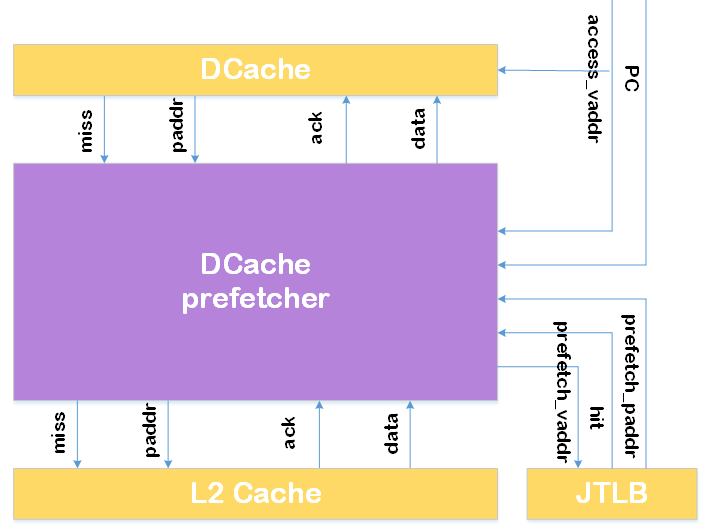
如果miss地址在两路buffer中都没有命中，则意味着当前有一个新的指令流产生。Miss地址会向L2 Cache发起数据请求，同时根据miss地址准备新的预取流。Demand request和prefetch request会同时送给一个arbiter，由arbiter来决定送给L2 cache的请求，每次只能发送一个请求，一个请求处理完成才可以发起下一个请求。Demand request的优先级永远要高于prefetch request。每检测到一个新的取指流都要flush掉一个旧的取指流，flush的路由两路乒乓产生。L2 cache取来的数据或者stream buffer提供的数据都是送给arbiter， 由arbiter来对ICache进行数据输送控制。可以把arbiter看作协调中枢。

取来送给stream buffer的指令会进行预译码，当检测到jump类指令时，对应流的预取停止，否则只要buffer有空，预取会一直进行（当然TLB miss时也会停止）。因为jump一定会修改指令流，这样可以避免不必要的预取占用带宽资源。

# DCache Prefetcher

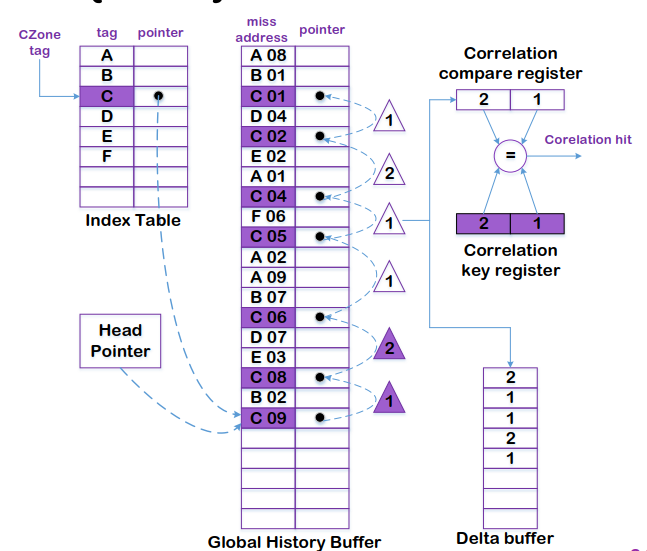
**（L2 Cache的实现包含L1 cache的数据，在L2 Cache实现中是否可以增加特殊标记，来标识line数据是否在L1 Cache中？这样DCache的预取地址可以不用查DCache来判断数据是否已经存在，DCache只需要把预取地址发给L2 Cache，L2 Cache如果检测到对应line已经存在L1中，便无效该预取操作）**

加入DCache prefetcher后，原有的DCache和L2 Cache之间的连接信号被截断，经由prefetcher和预取请求进行优先级处理后再送给L2 Cache。除此之外，我们使用指令的PC来标识访存地址所属数据流，记录的访存地址为虚拟line地址（虚拟地址可以进行跨页预取），采用line delta的方式来检测数据流的stride（这样只需要存储line地址即可，不用存储完整的miss虚拟地址，可以减少所使用存储面积）。预取器计算出来的candidate仍然是虚拟地址，使用该地址访问JTLB转化为物理地址，如果JTLB miss，则对应地址的预取操作取消，预取操作不引起异常。使用经过JTLB转化的物理地址来对L2 Cache发起预取操作，prefetch request和demand request区分开来，demand request的优先级高，但每次只对L2 Cache发起一个请求，有专门的请求发起仲裁部件，L2 Cache看到的依然只有一个请求信号和请求的物理地址。不论是prefetch request还是demand request，取来的数据都是直接放入DCache中。

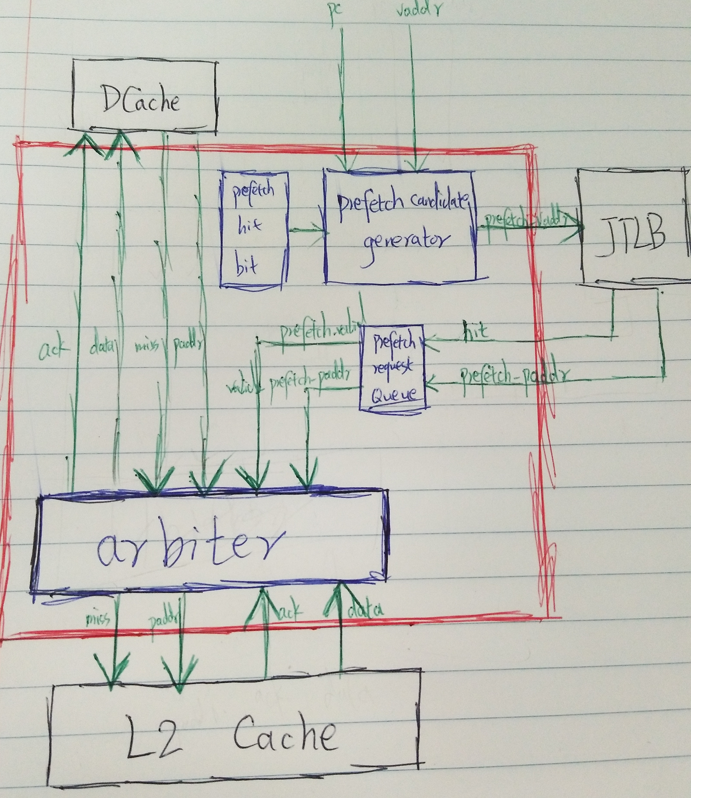


因为有预取，如果预取的发起条件只是miss的话，检测到的pattern模式会有缺失，所以要使用miss和prefetch hit地址一起构建pattern，需要为DCache中的每个line增加1bit的prefetch标识位，reset时该位为0，demand request拿来的数据该位也为0，只有prefetch request拿来的数据该位置1，当prefetch来的数据被真正访问时，清0。

对于line delta的模式检测机制，我们使用GHB的结构，其中IT的索引为load指令的PC，GHB中保存的是DCache访问的miss或者prefetch hit的line地址，具体的delta检测使用下面结构类似的方式：

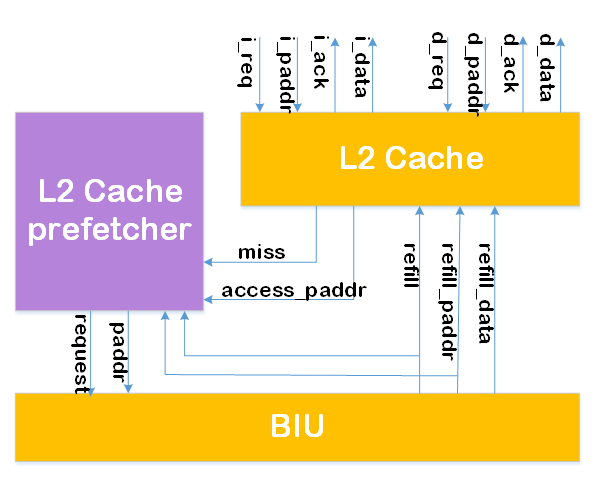


下图是DCache prefetcher的内部结构示意图：其中prefetch hit bit结构是用来记录DCache预取line是否被后续使用命中，会根据DCache的访问和重填信息进行更新。Prefetch candidate generator是一个GHB结构，根据指令PC和miss或者prefetch hit的访存虚地址对GHB进行更新并计算出预取虚拟地址。计算出来的虚拟地址要访问JTLB转化为物理地址，转化后的物理地址会按顺序存入prefetch request queue。Prefetch\_reuest和demand\_request会经过arbiter进行选择，总是优先demand\_request的处理。

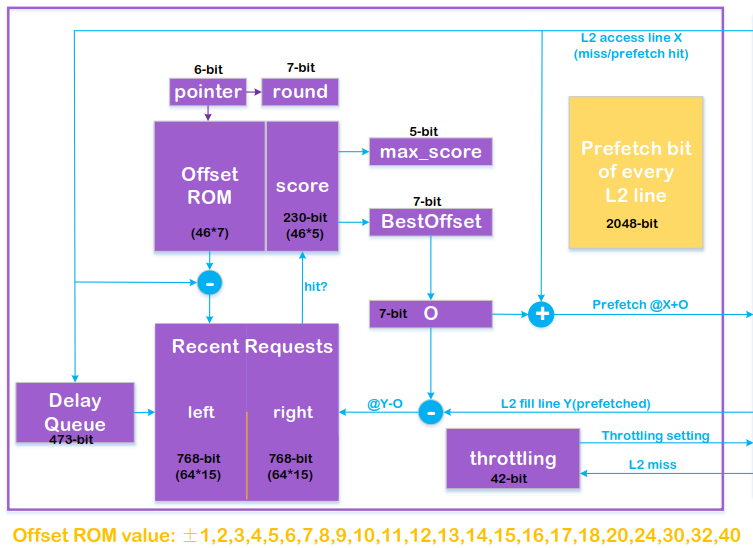


# L2 Cache prefetcher

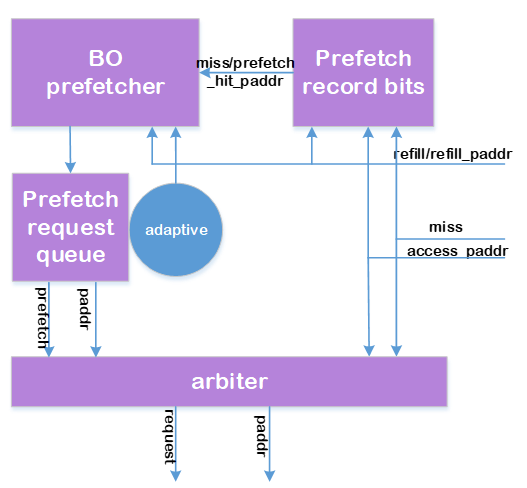
下面是L2 Cache prefetcher的接口示意图。同样L2 Cache向BIU的请求信号经由L2 Cache Prefetcher后再送给下一级。L2 Cache prefetcher根据L2 Cache的访问地址以及miss或者prefetch hit来产生预取地址。Prefetch request会和demand request一样送给BIU取数据，但是demand request的优先级要高于任何prefetch request。



L2 Cache prefetcher的核心是一个Best Offset Prefetcher，它会通过阶段性学习为下一阶段提供唯一的一个最佳offset进行预取，所有的miss或者prefetch hit都会加上该offset来产生预取地址。Miss和prefetch hit地址除了启动预取外，也用于本阶段的offset学习。下图是BO prefetcher的工作原理示意图。



Refill到L2 Cache的地址会被记录到Recent Request Table中，miss或者prefetch hit发生时将地址存入RR Table 时预取的timeliness会比较好；而refill时再将引起refill的miss或者prefetch hit地址存入RR Table的话，coverage和accuracy会比较好。通过对miss地址和RR之间增加Delay Queue，我们可以通过调整Delay Queue来获取timeliness和coverage之间的比较好的平衡点从而获取最佳预取性能。（如果DCache预取打开的话，只是用refill地址的更新应该就可以）



上图是L2 Cache prefetcher的内部结构示意图。BO prefetcher产生的预取请求会存入prefetch request queue，队列里的预取请求会和L2 Cache的miss请求一起送入arbiter，由arbiter来选择要发送给BIU的请求。BIU反馈回来的数据会直接送到L2 Cache中。

因为对于有些测试程序预取会降低性能，所以考虑加入一个adaptive器件来调整预取的激进程度。它可以采集的信息有预取请求队列的使用情况，L2 miss的频率，预取的coverage和accuracy等信息。