[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 12/56 (2006.01)

[12] 发明专利说明书

专利号 ZL 200510126361.1

[45] 授权公告日 2008 年 7 月 23 日

[11] 授权公告号 CN 100405786C

[22] 申请日 2005.12.9

[21] 申请号 200510126361.1

[73] 专利权人 清华大学

地址 100084 北京市北京 100084 - 82 信箱

[72] 发明人 胡成臣 刘 斌 陈雪飞 陈洪明

[56] 参考文献

US6219728 B1 2001.4.17

WO02098153A1 2002.12.5

US6717912B1 2004.4.6

US5901147A 1999.5.4

WO0005656A1 2000.2.3

Dynamic Queue Length Thresholds for Shared - MemoryPacket Switches. Abhijit K. Choudhury, Ellen L. Hahne. IEEE/ACM TRANSACTIONS ON NETWORKING, Vol. 6 No. 2. 1998

Optimal Buffer sharing. Israel Cidon. IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATION, Vol. 13 No. 7. 1995

Random Early Detection Gateways for Congestion Avoidance. Sally Floyd, Van Jacobson. IEEE/ACM TRANSACTIONS ON NETWORKING, Vol. 1 No. 4. 1993

审查员 张 畅

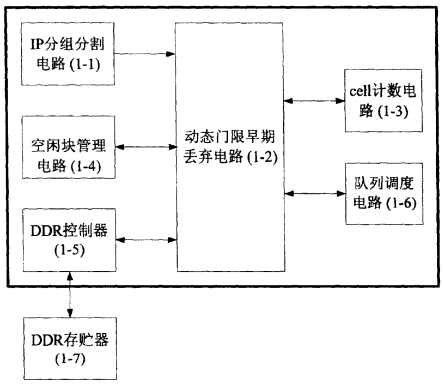
权利要求书 3 页 说明书 10 页 附图 7 页

[54] 发明名称

支持多队列的共享缓存动态门限早期丢弃装置

[57] 摘要

本发明提供一种支持多队列的共享缓存动态门限早期丢弃装置属于 IP 技术领域，并在一片现场可编程门阵列(FPGA)上实现，其特征在于：含有如图 1 的电路：IP 分组分割电路(1 - 1)；动态门限早期丢弃电路(1 - 2)；cell 计数电路(1 - 3)；空闲块管理电路(1 - 4)；DDR 控制器(1 - 5)；队列调度电路(1 - 6)；片外 DDR 存储器(1 - 7)。它能根据每个当前活跃的队列的平均队列长度和整个共享缓存区的平均队列长度来动态调整随机早期检测(RED)算法的参数，提出了支持多队列的共享缓存动态门限早期丢弃方法，其丢包率更小，缓存利用率更高，同时兼顾公平性。支持多队列的共享缓存动态门限早期丢弃方法保持了 RED 和动态门限(DT)机制的优点，并且用阶梯式丢弃曲线近似，利于在 FPGA 中实现。



1、支持多队列的共享缓存动态门限早期丢弃装置，其特征在于，该装置是用 FPGA 芯片实现的，该 FPGA 芯片中含有：IP 分组分割电路、空闲块管理电路、片外双倍数据速率存储器控制器、流单元计数电路、队列调度电路以及动态门限早期丢弃电路，其中：

IP 分组分割电路的输出端与动态门限早期丢弃电路的输入端相连，空闲块管理电路与动态门限早期丢弃电路互连，片外双倍数据速率存储器控制器与动态门限早期丢弃电路互连，流单元计数电路与动态门限早期丢弃电路互连，队列调度电路动态门限早期丢弃电路互连；

IP 分组分割电路，该电路把到达的变长的 IP 分组按设定的流单元长度进行分割，得到定长的流单元，用 cell 表示，所述 IP 分组分割电路含有：

第 1 个先进先出存储器设有 IP 分组输入端；

计数器，该计数器的计数信号输入端与所述第 1 个先进先出存储器读 FIFO 计数输出端相连；

分路器，该分路器有两个输入端：一个输入端与所述第 1 个先进先出存储器的 IP 数据输出端相连，该分路器的另一个输入端与所述计数器的计数输出端相连；

IP 包头信息寄存器，该寄存器的 IP 包头信息输入端与所述分路器的 IP 包头信息输出端相连；

流单元头寄存器，设有流单元输入端，该输入端与所述 IP 包头信息寄存器的相应输出端相连；

选择器内设有预定的流单元长度，所述选择器的流单元头信息输入端、IP 数据输入端依次分别与所述流单元头寄存器、分路器的 IP 数据输出相连；

流单元数据寄存器，该寄存器的流单元数据输入端与所述选择器的相应输出端相连；

第 2 个先进先出存储器，该存储器的流单元数据输入端与所述流单元数据寄存器的相应输出端相连，该先进先出存储器输出分割得到的流单元；

流单元计数电路，含有：加减计数器和磁性随机存取存储器，所述的加减计数器设有：流单元接纳指示信号输入端，接收片外双倍数据速率存储器的流单元；流单元调度指示信号输入端，接收流单元的调度信号；先前流单元数目输入端；所述的磁性随机存取存储器，用 MRAM 表示，设有：流单元的流号输入端；当前流单元数目输入端，该输入端与所述加减计数器的当前流单元数目输出端相连；该 MRAM 还有：先前流单元数输出端，该输出端与所述加减计数器的先前流单元数目输入端相连；该加减计数器在当一个流单元被接纳加入队列时计数器加 1，当一个流单元被调出离开队列时计数器减 1，该加减计数器内还设有队列权重的值 W_q，按下式计算 t 时刻的平均队列长度 $L^i_{avg}(t)$ 输出：

$$L^i_{avg}(t) = (1 - W_q)L^i_{avg}(t_{old}) + W_q Q^i(t)$$

其中，$L^i_{avg}(t_{old})$ 为 t 时刻队列 i 先前的流单元数；

$Q^i(t)$ 为 t 时刻队列 i 到达的流单元数目，该加减计数器同时设有一个 $Q(t)$ 输出端，$Q(t)$

是指 t 时刻所有队列长度总和，$Q(t)=\Sigma_i Q^i(t)$；

空闲块管理电路，该电路是一个空闲块加减计数器，该计数器内设有片外双倍数据速率存储器的容量，该加减计数器还设有流单元接纳指示信号输入端和流单元调度指示信号输入端，当一个流单元被接纳加入队列，或者一个流单元被调出队列时，该加减计数器依次分别对设定的片外双倍数据速率存储器的空闲容量减 1 或者加 1，据此输出该片双倍数据速率存储器的空闲容量；

片外双倍数据速率存储器的控制器，该控制器连接着一个所述的片外双倍数据速率存储器，对该存储器的存取访问进行控制；

队列调度电路，该电路设有对所述片外双倍数据速率存储器中的流单元进行读入调度的队列调度信号输出端；

动态门限早期丢弃电路，该电路设有：$L'_{avg}(t)$ 寄存器、进行丢弃控制的队列长度的下限阈值 $L_{\min}(t)$ 运算的第一运算器、进行丢弃控制的队列长度的上限阈值 $L_{\max}(t)$ 运算的第二运算器、比较器和进行丢弃控制的第三运算器，其中，

第一运算器，内置有 $L_{\min}(t)$ 冗余度，用 α 表示，还置有所述的片外双倍数据速率存储器的容量，用 B 表示，单位是流单元长度，该运算器通过 $Q(t)$ 信号输入端接收 $Q(t)$ 信号并按下式计算 $L_{\min}(t)$：

$L_{min}(t)=\alpha(B-Q(t))$；

第二运算器，内置有 $L_{\max}(t)$ 冗余度，用 β 表示，$\beta>\alpha$，还置有所述的片外双倍数据速率存储器的容量，用 B 表示，单位是流单元长度，该运算器通过 $Q(t)$ 信号输入端接收 $Q(t)$ 信号并按下式计算 $L_{\max}(t)$：

$L_{\max}(t)=\beta(B-Q(t))$

上述各 $Q(t)$ 信号输入端与所述流单元计数电路内的加减计数器的 $Q(t)$ 输出端相连；

比较器，内置有最大丢弃概率 $p_{\max}$，且该比较器设有：$L'_{avg}(t)$ 信号输入端，该输入端与所述流单元计数电路内的加减计数器的当前流单元数目输出端相连，该比较器还设有 $L_{\min}(t)$ 信号和 $L_{\max}(t)$ 信号共两个输入端，该比较器在接收到 $L'_{avg}(t)$、$L_{\min}(t)$、$L_{\max}(t)$ 信号后，依次按以下步骤执行：

步骤 1：若 $L'_{avg}(t)\le L_{min}(t)$，则接收全部到达的分组，并输出到达的所有流单元，丢弃值为 0；

步骤 2：若 $L_{\min}(t)<L'_{avg}(t)<L_{\max}(t)$ 时，则计算丢弃概率 p_a，并以此概率丢弃到达分组：

$$p_b=p_{\max}\frac{(L'_{avg}(t)-L_{\min}(t))}{(L_{\max}(t)-L_{\min}(t))}$$

$p_a=p_b/(1-cp_b)$，设 c=-1；

步骤 3：若 $L'_{avg}(t) \geq L_{\max}(t)$，则丢弃全部分组，c=0；

第三运算器，该运算器按所述比较器输出的流单元丢弃标志，即所述 c 的值按下式计算丢弃概率 p_a：

c=-1，则 $p_a = p_b/(1+p_b)$，丢弃部分到达的分组；

c=0，则 $p_a = p_b$，丢弃所有到达的分组。

支持多队列的共享缓存动态门限早期丢弃装置

技术领域

本发明属于 IP 技术领域。

背景技术

当前 Internet 中间节点设备，如路由器等，通过对其内部的缓存队列长度的控制，来影响和控制网络的拥塞情况。目前见诸于设备的传统的缓存队列管理机制是尾部丢弃(Tail Drop)机制：对每个队列设置一个长度门限，如果队列长度没有到达设定门限，接收所有分组进入队列；否则丢弃到达的分组。该机制实现简单，但是存在三个严重的缺陷：

（1）持续的满队列状态（Full Queues）；

（2）业务流对缓存的死锁（Lock Out）；

（3）业务量的全局同步(Global Synchronization)；

目前普遍认可的方法是加入中间节点的增强功能：主动队列管理（AQM：Active Queue Management)。即中间节点需要在网络进入拥塞情况之前，预先丢弃部分分组，以使得 TCP 协议响应来减慢源端的发送速率，避免拥塞的发生。在现有的网络设备中被采用的 AQM 机制是随机早期检测（RED：Random Early Detection）方法。但是该机制的主要缺点是：

（1）参数的设定困难，而且其性能敏感于参数和网络情况的变化；

（2）需要对每个单队列设置参数和计算，不利于队列数目或者网络流数目扩展的情况；

（3）RED 算法中有很多乘法运算，需要大量资源，很难用硬件高速实现。

发明内容

本发明的目的在于从硬件的角度提出一种能够进行高速处理的支持多队列的共享缓存动态门限早期丢弃装置。与目前普遍采用的随机早期检测（RED）机制相比，本发明包含以下几个部分的特点和内容：

（1）动态调整进行分组丢弃的队列门限值方法。原始的 RED 算法需设定一对最大、最小队列长度的门限的参数，当队列状态处于这一对门限的范围中时，按照一定的概率丢弃到达分组。由于参数是预先设定的，很难满足多变的网络状况，因而性能会随着设定参数和网络情况的变化而变化。本发明克服 RED 算法对于参数设置和网络情况的依赖，动态调整进行分组丢弃的一对门限，即根据每个当前活跃的队列的平均队列长度和整个共享缓存区的平均队列长度来动态调整丢弃控制的阈值。这一机制的基本思想是：当网络拥塞状况的趋势增加，即剩余缓存空间减少时，提前进入分组丢弃的控制阶段；而当网络拥塞状况的趋势减弱，即剩余缓存空间增加时，推迟进入分组丢弃的控制阶段。

（2）采用阶梯式丢弃曲线方法在硬件中实现分组的丢弃。在动态确定分组丢弃的范围

之后，需要对进入丢弃控制范围内的到达流的分组进行部分丢弃。对于越靠近最大丢弃门限的流，将丢弃越多的到达分组，相对地，对于越靠近最小丢弃门限的流，将丢弃越少的到达分组。而当该网络流超过最大丢弃门限时，完全丢弃到达分组。对于位于丢弃范围内网络流的队列，我们采用如图 3 所示的阶梯式丢弃曲线来计算应该丢弃的分组的比例，从而避免浮点了运算，使得能够用硬件查表的方式高速实现。

（3）在共享缓存上的多队列管理机制。随着目前 internet 网络流的数目的增加和对各种网络流的服务质量要求的提高，在网络中间节点设备中也要求能够支持大量的网络流的单独排队和管理。传统的方法是事先设定好队列数目和各个队列的长度。这种方法缓存利用率低，而且不利于队列数目的扩展。本发明在共享缓存上进行不同分组的缓存。在网络流到达或者退出中间节点设备时，动态生成或者撤销队列。已建立队列的网络流的分组到达时，采用（1）、（2）中所述的方法，判断是否接纳该分组。是，则从空闲的缓存中分配空间连接到其所述的队列中；否，则直接丢弃。

支持多队列的共享缓存动态门限早期丢弃装置，其特征在于，该装置是用 FPGA 芯片实现的，该 FPGA 芯片中含有：IP 分组分割电路、空闲块管理电路、片外双倍数据速率存储器控制器、流单元计数电路、队列调度电路以及动态门限早期丢弃电路，其中：

IP 分组分割电路的输出与动态门限早期丢弃电路相连，空闲块管理电路的输入输出与动态门限早期丢弃电路相连，片外双倍数据速率存储器控制器的输入输出与动态门限早期丢弃电路相连，流单元计数电路的输入输出与动态门限早期丢弃电路相连，队列调度电路的输入输出与动态门限早期丢弃电路相连。

IP 分组分割电路，该电路把到达的变长的 IP 分组按设定的流单元长度进行分割，得到定长的流单元，用 cell 表示，所述 IP 分组分割电路含有：

第 1 个先进先出存储器设有 IP 分组输入端；

计数器，该计数器的计数信号输入端与所述第 1 个先进先出存储器读 FIFO 计数输出端相连；

分路器，该分路器有两个输入端：一个输入端与所述第 1 个先进先出存储器的 IP 数据输出端相连，该分路器的另一个输入端与所述计数器的计数输出端相连；

IP 包头信息寄存器，该寄存器的 IP 包头信息输入端与所述分路器的 IP 包头信息输出端相连；

流单元头寄存器，设有流单元输入端，该输入端与所述 IP 包头信息寄存器的相应输出端相连；

选择器内设有预定的流单元长度，所述选择器的流单元头信息输入端、IP 数据输入端依次分别与所述流单元头寄存器、分路器的 IP 数据输出相连；

流单元数据寄存器，该寄存器的流单元数据输入端与所述选择器的相应输出端相连；

第 2 个先进先出存储器，该存储器的流单元数据输入端与所述流单元数据寄存器的相应输出端相连，该先进先出存储器输出分割得到的流单元；

流单元计数电路，含有：加减计数器和磁性随机存取存储器，所述的加减计数器设有：流单元接纳指示信号输入端，接收片外双倍数据速率存储器的流单元；流单元调度指示信号输入端，接收流单元的调度信号；先前流单元数目输入端；所述的磁性随机存取存储器，用MRAM表示，设有：流单元的流号输入端；当前流单元数目输入端，该输入端与所述加减计数器的当前流单元数目输出端相连；该MRAM还有：先前流单元数输出端，该输出端与所述加减计数器的先前流单元数目输入端相连；该加减计数器在当一个流单元被接纳加入队列时计数器加 1，当一个流单元被调出离开队列时计数器减 1，该加减计数器内还设有队列权重的值 W_q，按下式计算 t 时刻的平均队列长度 $L^i_{avg}(t)$ 输出：

$$L^i_{avg}(t) = (1 - W_q)L^i_{avg}(t_{old}) + W_q Q^i(t)$$

其中，$L^i_{avg}(t_{old})$ 为 t 时刻队列 i 先前的流单元数；

$Q^i(t)$ 为 t 时刻队列 i 到达的流单元数目，该加减计数器同时设有一个 $Q(t)$ 输出端，$Q(t)$ 是指 t 时刻所有队列长度总和，$Q(t) = \sum_i Q^i(t)$；

空闲块管理电路，该电路是一个空闲块加减计数器，该计数器内设有片外双倍数据速率存储器的容量，该加减计数器还设有流单元接纳指示信号输入端和流单元调度指示信号输入端，当一个流单元被接纳加入队列，或者一个流单元被调出队列时，该加减计数器依次分别对设定的片外双倍数据速率存储器的空闲容量减 1 或者加 1，据此输出该片双倍数据速率存储器的空闲容量；

片外双倍数据速率存储器的控制器，该控制器连接着一个所述的片外双倍数据速率存储器，对该存储器的存取访问进行控制；

队列调度电路，该电路设有对所述片外双倍数据速率存储器中的流单元进行读入调度的队列调度信号输出端；

动态门限早期丢弃电路，该电路设有：$L^i_{avg}(t)$ 寄存器、进行丢弃控制的队列长度的下限阈值 $L_{\min}(t)$ 运算的第一运算器、进行丢弃控制的队列长度的上限阈值 $L_{\max}(t)$ 运算的第二运算器、比较器和进行丢弃控制的第三运算器，其中，

第一运算器，内置有 $L_{\min}(t)$ 冗余度，用 α 表示，还置有所述的片外双倍数据速率存储器的容量，用 B 表示，单位是流单元长度，该运算器通过 $Q(t)$ 信号输入端接收 $Q(t)$ 信号并按下式计算 $L_{\min}(t)$：

$$L_{\min}(t) = \alpha(B - Q(t))$$；

第二运算器，内置有 $L_{\max}(t)$ 冗余度，用 β 表示，$\beta > \alpha$，还置有所述的片外双倍数据速率存储器的容量，用 B 表示，单位是流单元长度，该运算器通过 $Q(t)$ 信号输入端接收 $Q(t)$ 信号并按下式计算 $L_{\max}(t)$：

$$L_{\max}(t) = \beta(B - Q(t))$$

上述各 $Q(t)$ 信号输入端与所述流单元计数电路内的加减计数器的 $Q(t)$ 输出端相连；

比较器，内置有最大丢弃概率 p_{max}，且该比较器设有：$L'_{avg}(t)$ 信号输入端，该输入端与所述流单元计数电路内的加减计数器的当前流单元数目输出端相连，该比较器还设有 $L_{min}(t)$ 信号和 $L_{max}(t)$ 信号共两个输入端，该比较器在接收到 $L'_{avg}(t)$、$L_{min}(t)$、$L_{max}(t)$ 信号后，依次按以下步骤执行：

步骤 1：若 $L'_{avg}(t) \le L_{min}(t)$，则接收全部到达的分组，并输出到达的所有流单元，丢弃值为 0；

步骤 2：若 $L_{min}(t) < L'_{avg}(t) < L_{max}(t)$ 时，则计算丢弃概率 p_a ，并以此概率丢弃到达分组：

$$p_b = p_{max} \frac{(L'_{avg}(t) - L_{min}(t))}{(L_{max}(t) - L_{min}(t))}$$

$p_a = p_b/(1 - cp_b)$，设 c=-1；

步骤 3：若 $L'_{avg}(t) \ge L_{max}(t)$，则丢弃全部分组，c=0；

第三运算器，该运算器按所述比较器输出的流单元丢弃标志，即所述 c 的值按下式计算丢弃概率 p_a：

c=-1，则 $p_a = p_b/(1 + p_b)$，丢弃部分到达的分组；

c=0，则 $p_a = p_b$，丢弃所有到达的分组。

本发通过测试证明，具有适应性强、缓存利用率高而且支持多流队列动态管理机制的优点。

附图说明

图 1： 基本结构模块和外部接口关系

图 2： RED 算法丢弃曲线

图 3： 本发明的丢弃机制的阶梯式丢弃曲线

图 4： 硬件实现流程图

图 5： IP 分组分割电路（1-1）

图 6： 动态门限早期丢弃电路（1-2）

图 7： cell 计数电路（1-3）

图 8： 空闲块管理电路（1-4）

图 9： DDR 控制器（1-5）

图 10：队列调度电路（1-6）

图 11：第二段丢弃曲线波形(a)

图 12：第二段丢弃曲线波形(b)

图 13：第三段丢弃曲线波形(a)

图 14：第三段丢弃曲线波形(b)

图 15：第四段丢弃曲线波形(a)

图 16：第四段丢弃曲线波形(b)

具体实施方式

互联网工程任务组（IETF）提出了 AQM 技术并推荐了 RED 机制。实现 RED 算法的路由器发现拥塞前兆时，提前随机丢弃缓冲队列中的一些分组，而不是等到缓冲区占满后丢弃所有新的分组。当网络中间节点设备的缓存的平均队列长度超过一个指定的最小门限 $\min_{th}$ 时，就认为出现了拥塞前兆，这时路由器按一定的概率 p_a 丢弃分组，这个概率 p_a 是平均队列长度 $avg(t)$ 的函数：

$$p_b = p_{\max}\frac{(avg(t)-\min_{th})}{(\max_{th}-\min_{th})}$$

$$p_a = p_b/(1-cp_b)$$

其中 $p_{\max}$ 为设定的最大丢弃概率，c 的取值为一常数。

当平均队列长度超过一个指定的最大门限 $\max_{th}$ 时，路由器认为网络出现了严重拥塞，所有的分组都要丢弃。RED 算法丢弃曲线如图 2 所示。

从上述 RED 算法的介绍可知，硬件实现 RED 算法时存在以下几个问题：

(1)参数的选择：$\min_{th}$，$\max_{th}$，$\max_p$ 和 w_q 等参数的选择对 RED 算法有很大影响，但是这些参数是事先设置好的，不能实时反映当前网络的状况；

(2) RED 算法中有很多乘法运算，另外还需要生成随机数，用硬件高速实现有一定的困难；

(3) RED 算法采用先进先出的方法，对复用在一个接收队列中的所有连接的分组进行调度，但不监视每个流的状态，因此不支持多流多队列。

本发明在一片 FPGA 上，用硬件描述语言 Verilog HDL 编写实现，其基本结构和外部接口关系如图 1 所示，各电路工作流程为：当一个 IP 分组到达该装置时，由 IP 分组分割电路(1-1)切分为 60 字节的定长的数据单元，称作 cell，然后将其发送给动态门限早期丢弃电路(1-2)。动态门限早期丢弃电路（1-2）模块根据动态门限早期丢弃方法的策略对 cell 实施接纳和丢弃控制，被接纳的 cell 再加上通过 DDR 控制器（1-5）写入片外 DDR 存储器（1-7）相应的网络流缓存队列中，等待队列调度电路（1-6）的调度命令。队列调度电路（1-6）对片外的 DDR 存储器中的 cell 进行调度，从众多的队列中选择某个流的队列，调度 DDR 存储器中的分组离开。cell 计数电路（1-3）按照 cell 的流号，对 DDR 中的 cell 进行计数，对每个流都维护一个计数器，当一个 cell 接纳进入队列时计数器加一，当一个 cell 被调度离开队列时计数器减一。空闲块管理电路（1-4）负责统计片外 DDR 中的空闲容量，当一个 cell 被接纳进入队列时，分配一个空闲块，并且其数目减一；当一个 cell 被调度离开队列时，回收一个空闲块，并且其数目加一。动态门限早期丢弃电路（1-2）依据 cell 计数电路（1-3）提供的片外

DDR 存储器中该流的当前 cell 数目和空闲块管理电路（1-4）提供的片外 DDR 中的当前空闲容量两个参数对 cell 实施接纳和丢弃控制。

我们规定，以$Q^i(t)$表示 t 时刻第 i 个等待队列的长度；以$Q(t)=\Sigma_i Q^i(t)$表示 t 时刻所有队列长度总和，即缓存中被占用部分的大小；以 B 表示整个共享缓存区的大小；用$L_{\min}(t)$，$L_{\max}(t)$ 代表 t 时刻进行分组丢弃控制的队列长度的下限和上限阈值。

当每一个分组到达队列 i 时，计算 t 时刻队列 i 的平均队列长度$L^i_{avg}(t)$（W_q为设定的队列权重）：

$L^i_{avg}(t)=(1-W_q)L^i_{avg}(t_{old})+W_qQ^i(t)$	(1)

重新计算 t 时刻进行分组丢弃控制的队列长度的上限和下限阈值$L_{\min}(t)$，$L_{\max}(t)$（$\beta>\alpha$）：

$L_{\min}(t)=\alpha(B-Q(t))$	(2)

$$L_{\max}(t)=\beta(B-Q(t)) \quad (3)$$

根据计算得到的$L_{\min}(t)$，$L_{\max}(t)$，进行以下的判断：

（1）如果$L^i_{avg}(t)\le L_{\min}(t)$，不作任何控制，接收全部到达的分组；

（2）如果$L_{\min}(t)<L^i_{avg}(t)<L_{\max}(t)$时，则计算丢弃概率 p_a ，并以此概率丢弃到达分组。（$p_{\max}$为设定的最大丢弃概率）

$p_b=p_{\max}\dfrac{(L^i_{avg}(t)-L_{\min}(t))}{(L_{\max}(t)-L_{\min}(t))}$	(4)
$p_a=p_b/(1-cp_b),c=-1$	(5)

（3）如果$L^i_{avg}(t)\ge L_{\max}(t)$，则丢弃全部分组，c=0。

以下给出实现的伪代码：

初始化：$L^i_{avg}(t)$ = 0，c = - 1；

时刻 t，某一分组到达输入队列 i：

用式(1)计算$L^i_{avg}(t)$；

用式(2)、(3)计算$L_{\min}(t)$和$L_{\max}(t)$；

if $L_{\min}(t) < L'_{avg}(t) < L_{\max}(t)$

用公式(4)、(5)计算丢弃概率 p_a；

以概率 p_a 丢弃达到分组；

c=0;

else if $L'_{avg}(t) \geq L_{\max}(t)$

丢弃到达分组；

c=0;

else

c=－1;

考虑大小为 B 的缓存区中，当前只有 A 个队列处于活跃的状态。因为各个队列的长度被控制在 $L_{\max}(t)$ 左右，那么此时缓存区被占据的最大容量在 $Q(t) = AL_{\max}(t)$ 左右，代入式（3）中可以得到 $L_{\max}(t) = \frac{\beta B}{1+\beta A}$，所以缓存区的利用率为 $\rho = \frac{\beta A}{1+\beta A}$。从上式中可以看出，与 RED 算法一样，本发明也总是预留出一小部分缓存空间，能够更好地处理突发性。

在利用硬件高速实现时存在两个困难：一是分组丢弃控制门限 $L_{\min}(t)$ 和 $L_{\max}(t)$ 的计算；二是丢弃概率的计算。本发明通过对算法作以下的设置，就可以利用硬件来高速实现：

（1）每当有分组到达，由 cell 计数电路（1-3）给出该流的平均队列长度，记为 C;

（2）整个共享缓存区的大小 B 是所采用的片外 DDR 的大小；

（3）缓存中被占用部分的大小 $Q(t)$ 由空闲块管理电路（1-4）给出，记 DDR 中空闲缓存大小是 R;

（4）参数 $\alpha = 0.5$，$\beta = 1.0$，则由空闲块管理电路（1-4）给出的空闲缓存大小 R，$L_{\min}(t) = 0.5R$，$L_{\max}(t) = R$，其中通过简单的右移一位就可以得到分组丢弃控制门限 $L_{\min}(t)$。

由于丢弃是对一个完整的分组进行实施的，因此，在分组的第一个信元到来时，平均队列长度 C 和两个门限值 $L_{\min}(t)$，$L_{\max}(t)$ 进行比对，决定是丢弃该分组的 cell，还是将该分组的 cell 写入片外的 DDR 存储器。

另外，如果严格计算丢弃概率，并按照该概率随机丢弃到达的分组，需要复杂的伪随机数生成电路，并且有大量的乘法运算，会占用大量的 FPGA 资源，不利于用硬件高速实现。我

们可以采用如图 3 所示的分段函数逼近丢弃曲线，然后预先计算好分段门限值、剩余容量和丢弃概率之间的关系，实现时，仅需要通过从空闲块管理模块得到的 DDR 剩余容量 R 和从分组计数电路得到该流的当前队列长度 C，通过表 1 可查表得到分组丢弃概率，然后按照此丢弃概率周期性丢弃到达的分组。比如，当从分组计数电路得到的当前队列长度 $0 \le C < 0.5R$ 时，所有到达的分组都不丢弃，都通过 DDR 控制器（1-5）写入片外的 DDR 存储器；当 $0.5R \le C < 0.75R$ 时，该流每到达 8 个分组就丢弃一个分组。采用的分段函数逼近丢弃曲线方法以及周期性的丢弃规则，可在 FPGA 上高速实现，电路工作流程如图 4 所示。

表 1 剩余容量 R、流的队列长度 C 和分组丢弃概率的关系

分段号	当前队列长度 C	丢弃概率	丢弃规则
1	$0 \le C < 0.5R$	0%	不丢弃分组
2	$0.5R \le C < 0.75R$	12.5%	每 8 个分组丢弃一个
3	$0.75R \le C < R$	33.3%	每 3 个分组丢弃一个
4	$C \ge R$	100%	丢弃全部分组

基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置采用硬件描述语言 Verilog 在一片 FPGA 上实现，并在所开发的路由器线卡上实验和测试。测试系统由 PC 机的并行口，通过 FPGA 的 JTAG 口，向该线卡上的 FPGA 下载 HDL 代码，再用测试仪向目标板发送特定的 IP 分组，经 FPGA 上的模块处理之后，由测试仪和 EDA 工具对输出结果进行统计和记录，分析所设计的该装置正确性。

测试中用到的测试仪为 Spirent 公司的 AX/4000。AX/4000 是一种模块化的多端口测试系统，能够以高达 10Gbps 的速度同时测试 ATM、IP、帧中继和以太网等多种传输技术。AX/4000 测试仪采用模块化设计，主要模块由系统控制模块、发生器模块、分析器模块和丰富的接口模块组成。发生器模块提供 IP 源地址和目的地址、多种发包模式；分析器模块自动提取的流信息，实时地显示各流的丢包率、丢包统计等信息，并以数据表、线性图、直方图等方式显示。本专利的实验用 AX/4000 产生特定的流，并对结果进行统计和分析。

开发和测试中用的 EDA 工具主要为 Altera 公司的 Quartus 软件，Quartus 软件完成 Verilog 代码的编写、实现和下载，并用其中的 Signal Tap 分析和记录时序波形。

基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置采用阶梯式曲线来逼近 RED 的丢弃曲线，周期性地丢弃部分分组。由空闲块管理电路（1-4）得到的 DDR 剩余容量 R、由 cell 计数电路（1-3）给出流的队列长度 C 和分组丢弃概率的关系可由表 1 表示。基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置的实验要测试分段函数曲线的每一段。基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置中，如果队列调度电路（1-6）不进调度，该装置将会经历分段丢弃曲线的每一段，可在 Quartus 的 Signal Tap 中依次设置相应的触发值，

就可以记录和分析波形。

第一段曲线的测试：如果队列调度电路（1-6）正常调度，该装置工作在丢弃曲线的第一段，完全接纳到达的分组。在测试中，测试仪 AX/4000 发包模式均为 Manually Triggered Bursts，100%BW 发包，总的发包数为 300000，每次手动触发时发送 300000 个分组，再用测试仪 AX/4000 接收处理后的分组。从表 2 可以得出，该装置工作在丢弃曲的第一段时能正确转发分组，不会丢弃分组。

表 2 第一段丢弃曲线测试结果

序号	流数	发送的包数	包长	收到的包数	结论
1	1	300,000	40	300,000	正确
2	1	300,000	1500	300,000	正确
3	100	300,000	40	300,000	正确
4	100	300,000	1500	300,000	正确
5	1000	300,000	40	300,000	正确
6	1000	300,000	1500	300,000	正确

中间段的测试：如果队列调度电路（1-6）不进行调度，该装置将会经历分段丢弃曲线的每一段，依次在 Signal Tap 中设置相应的触发值，记录其波形。在测试中，测试仪 AX/4000 选择长度为 48 字节包，发包模式均为 periodic packets，100%BW。在测试中，在 Quartus 的 Signal Tap 中添加如下的信号：

1. C4_slotcycle:该装置的时钟节拍记数，计数值为 0~15；

2. F_first_cell：分组的第一个 cell 的标志；

3. CON_discard_interval：每段丢弃曲线的计数上限，比如在 $0.75R \le C < R$ 分段区间时，每 3 个分组丢弃一个，则 CON_discard_interval = 3；

4. C4_discard：在每段丢弃曲线的周期计数器，用于判断是否该丢弃该分组，比如在 $0.75R \le C < R$ 分段区间时，每 3 个分组丢弃一个，C4_discard 为 1~3 周期计数，当 C4_discard= CON_discard_interval = 3 时，丢弃该分组；

5. F_range_judge：在每段丢弃曲线的周期计数判断，当 C4_discard= CON_discard_interval 时，到达计数上限时，则 F_range_judge = 1，表示该分组要丢弃；

6. F_IP_discard：IP 分组丢弃标志，为 1 时，丢弃该分组，信号宽度为 IP 分组的宽度；

7. C19_Cellnum：当前流的在 DDR 中缓存的 cell 数，即队列长度 C；

8. C19_Remain：DDR 当前的剩余空间，即剩余容量 R；

9. F_range_show：RED-DT 所在丢弃曲线段的指示；

在 Quartus 的 Signal Tap 中设置 F_range_show 为触发条件，依次为：

00：第一段，不丢弃分组；

01：第二段，每 8 个分组丢弃一个；

10：第三段，每 3 个分组丢弃一个；

11：第四段，丢弃全部分组；

采用上述的测试方法，得到如下的结果：

（1）设置 F_range_shown 的触发值为 01，基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置工作在分段丢弃曲线的第二段，每 8 个分组丢弃一个分组，测试波形如图 11、图 12 所示。从图 11、图 12 可以看出，当 F_range_shown = 01 时，C = 172031，R = 344063，$0.5R \le C < 0.75R$，该装置工作在分段丢弃曲线的第二段，每 8 个分组丢弃一个分组，当 C4_discard = 8 时，F_IP_discard 和 F_range_judge 为 1，丢弃该分组，测试正确。

（2）设置 F_range_shown 的触发值为 10，基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置工作在分段丢弃曲线的第三段，每 3 个分组丢弃一个分组，测试波形如图 13、图 14 所示。从图 13、图 14 可以看出，当 F_range_shown = 10 时，C = 22183，R = 294911，$0.75R \le C < R$，该装置工作在分段丢弃曲线的第三段，每 3 个分组丢弃一个分组，当 C4_discard = 3 时，F_IP_discard 和 F_range_judge 为 1，丢弃该分组，测试正确。

（3）设置 F_range_shown 的触发值为 11，基于 FPGA 的支持多队列的共享缓存动态门限早期丢弃装置工作在分段丢弃曲线的第四段，丢弃全部分组，测试波形如图 15、图 16 所示。从图 15、图 16 可以看出，当 F_range_shown = 11 时，C = 258047，R = 258047，$C \ge R$，该装置工作在分段丢弃曲线的第四段，丢弃全部分组 F_IP_discard 和 F_range_judge 始终为 1，丢弃该分组，测试正确。

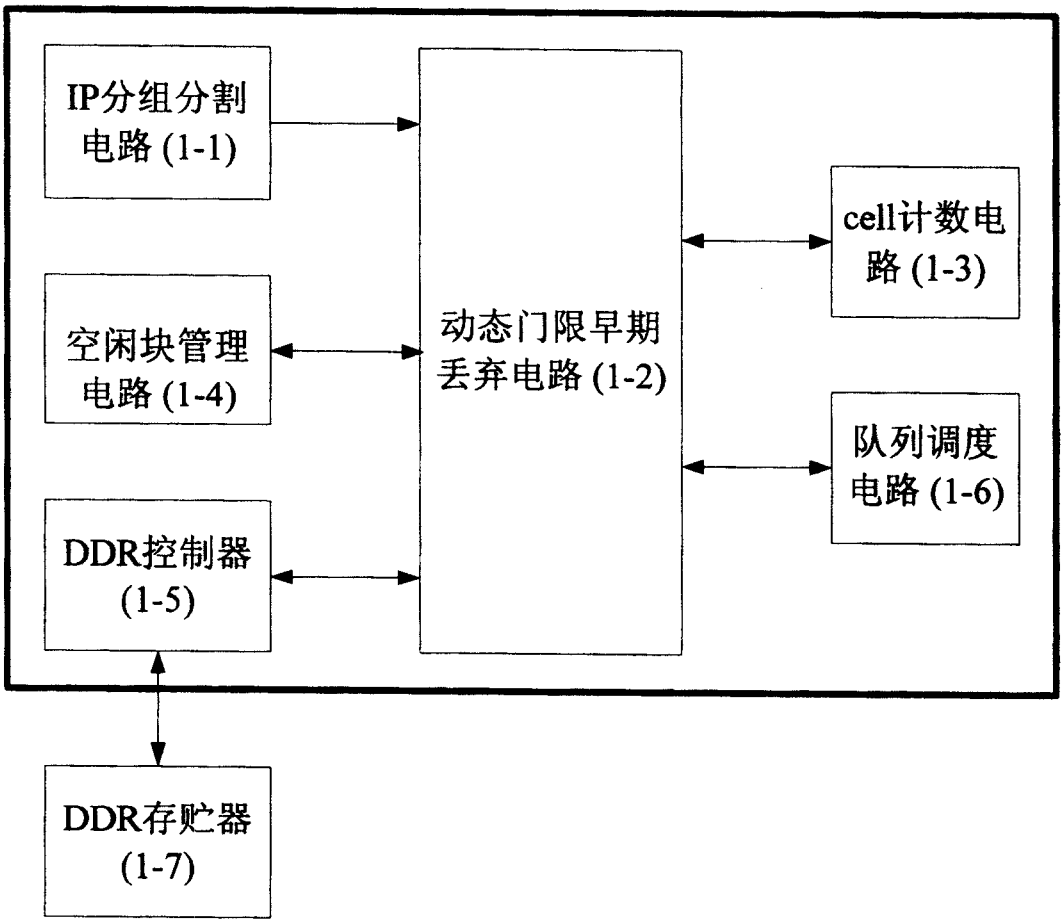

IP分组分割电路 (1-1)
空闲块管理电路 (1-4)
DDR控制器 (1-5)
DDR存贮器 (1-7)
动态门限早期丢弃电路 (1-2)
cell计数电路 (1-3)
队列调度电路 (1-6)

图 1

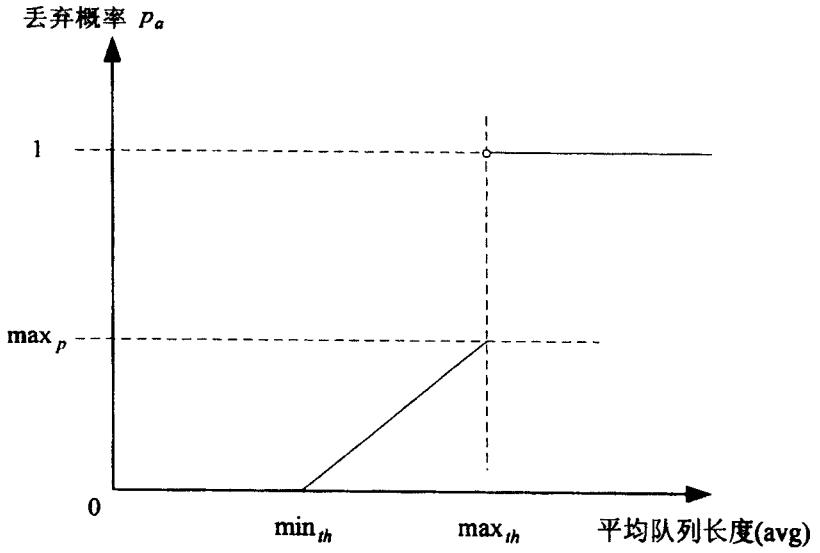

丢弃概率 p_a
1
$\max_p$
0
$\min_{th}$
$\max_{th}$
平均队列长度(avg)

图 2

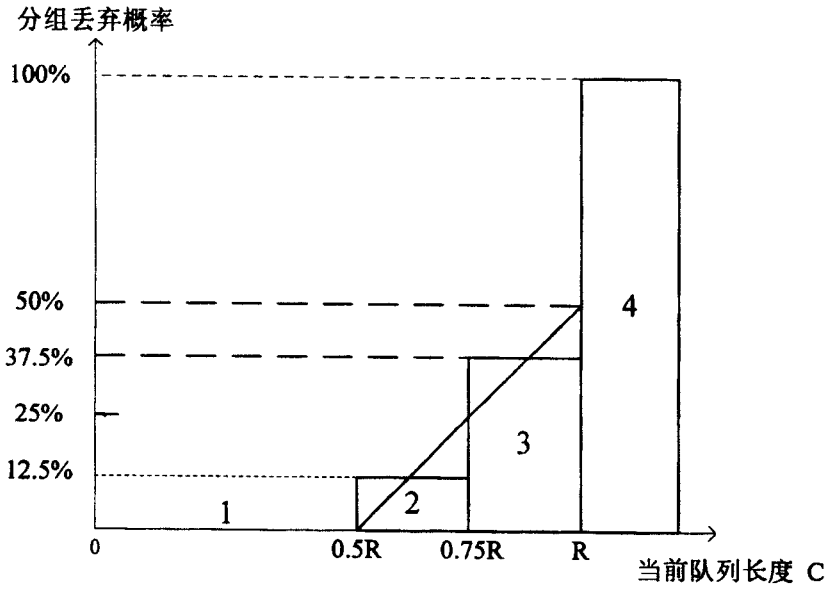

分组丢弃概率
100%
50%
37.5%
25%
12.5%
1
2
3
4
0
0.5R
0.75R
R
当前队列长度 C

图 3

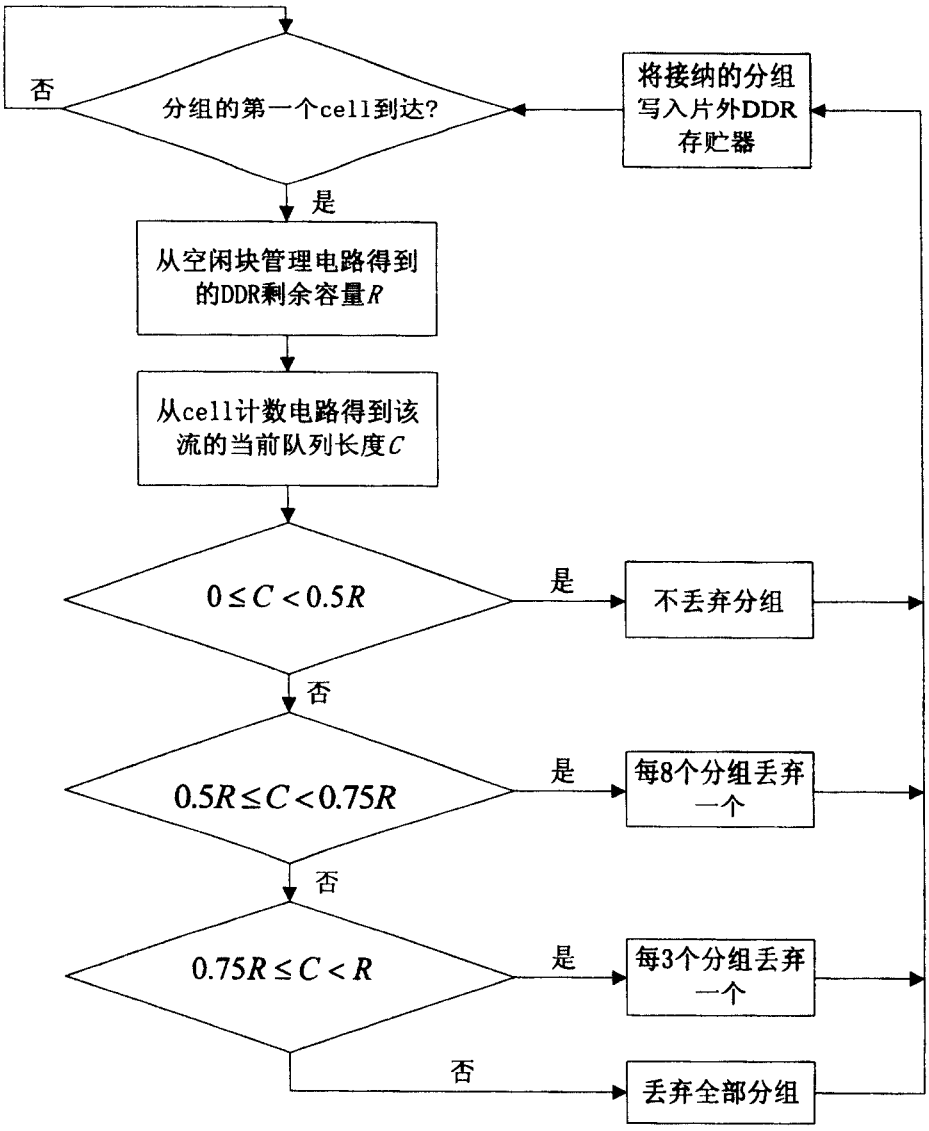

否
分组的第一个cell到达?
将接纳的分组写入片外DDR存贮器
是
从空闲块管理电路得到的DDR剩余容量R
从cell计数电路得到该流的当前队列长度C
0≤C<0.5R
是
不丢弃分组
否
0.5R≤C<0.75R
是
每8个分组丢弃一个
否
0.75R≤C<R
是
每3个分组丢弃一个
否
丢弃全部分组

图 4

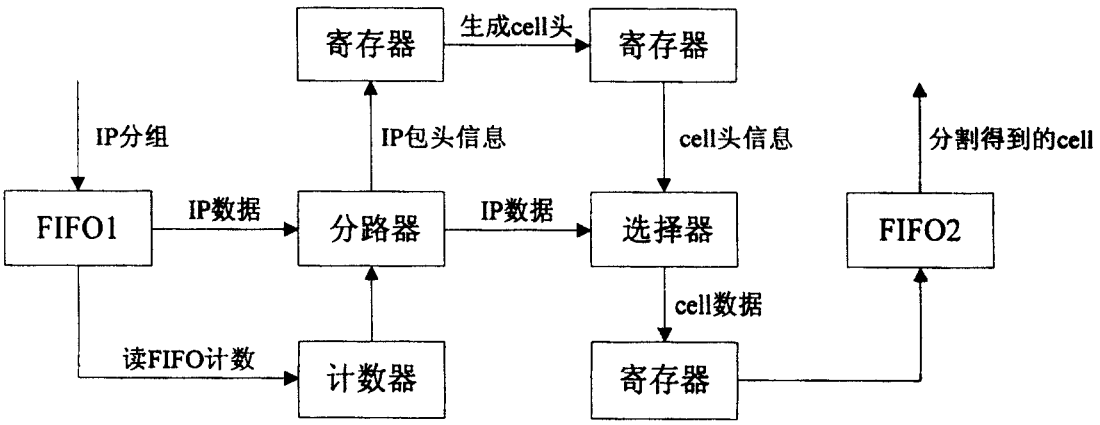

寄存器
生成cell头
寄存器
IP分组
IP包头信息
cell头信息
分割得到的cell
FIFO1
IP数据
分路器
IP数据
选择器
FIFO2
cell数据
读FIFO计数
计数器
寄存器

图 5

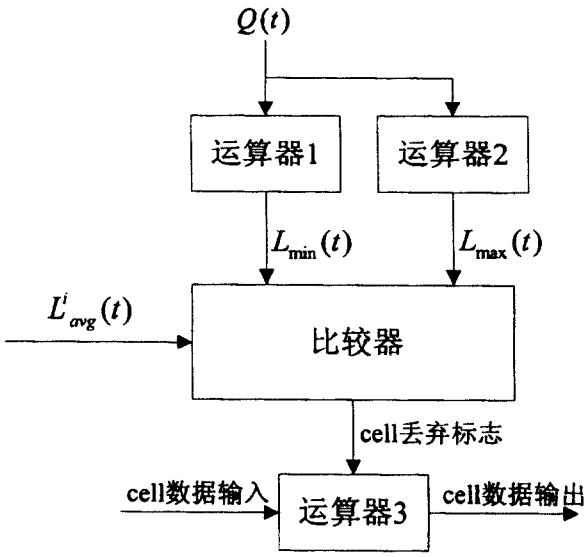

Q(t)
运算器1
运算器2
$L_{min}(t)$
$L_{max}(t)$
$L^i_{avg}(t)$
比较器
cell丢弃标志
cell数据输入
运算器3
cell数据输出

图 6

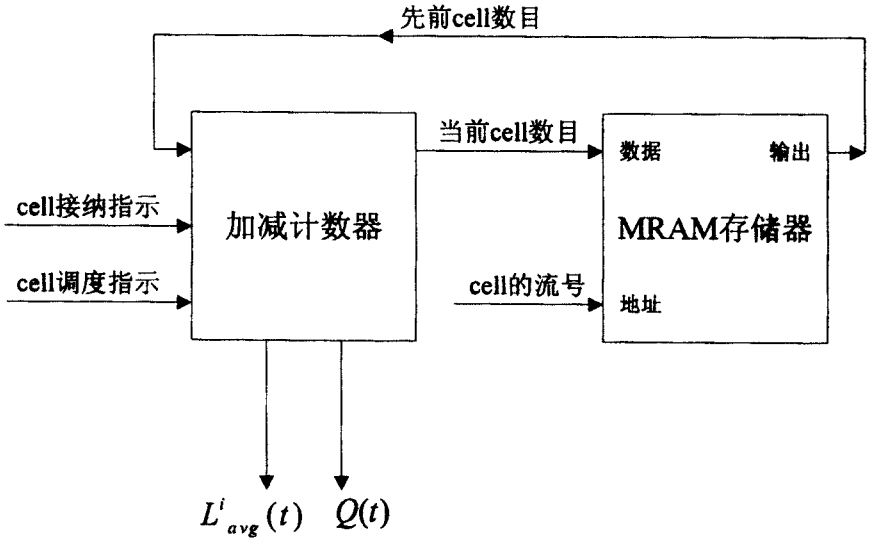

先前cell数目
当前cell数目
数据
输出
cell接纳指示
加减计数器
MRAM存储器
cell调度指示
cell的流号
地址
$L^i_{avg}(t)$
Q(t)

图 7

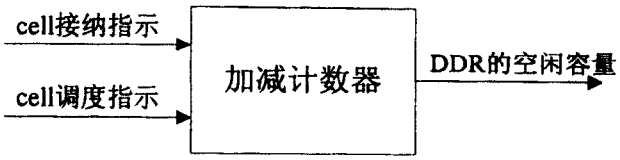

cell接纳指示
cell调度指示
加减计数器
DDR的空闲容量

图 8

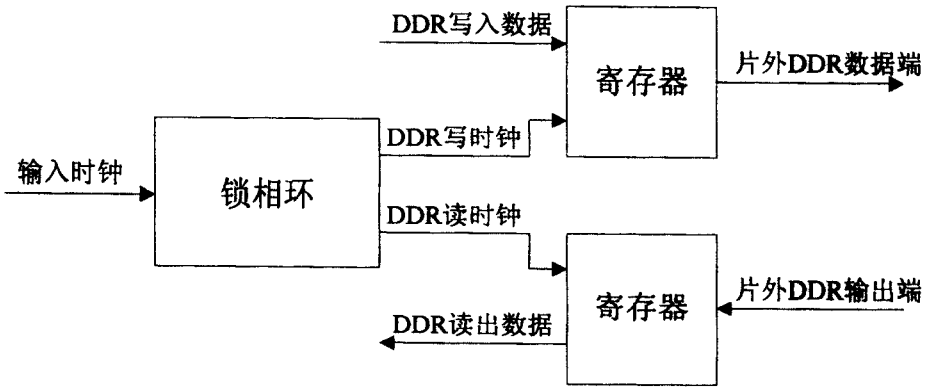

DDR写入数据
寄存器
片外DDR数据端
输入时钟
锁相环
DDR写时钟
DDR读时钟
寄存器
片外DDR输出端
DDR读出数据

图 9

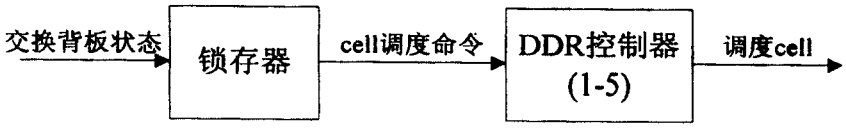

交换背板状态
锁存器
cell调度命令
DDR控制器
(1-5)
调度cell

图 10

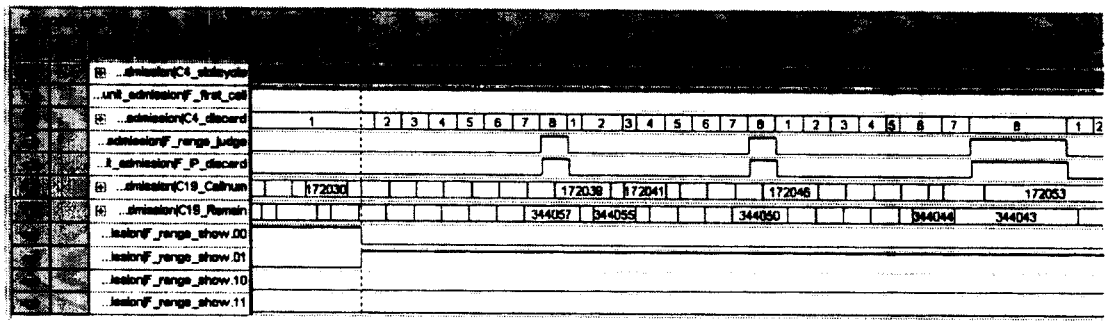


图 11

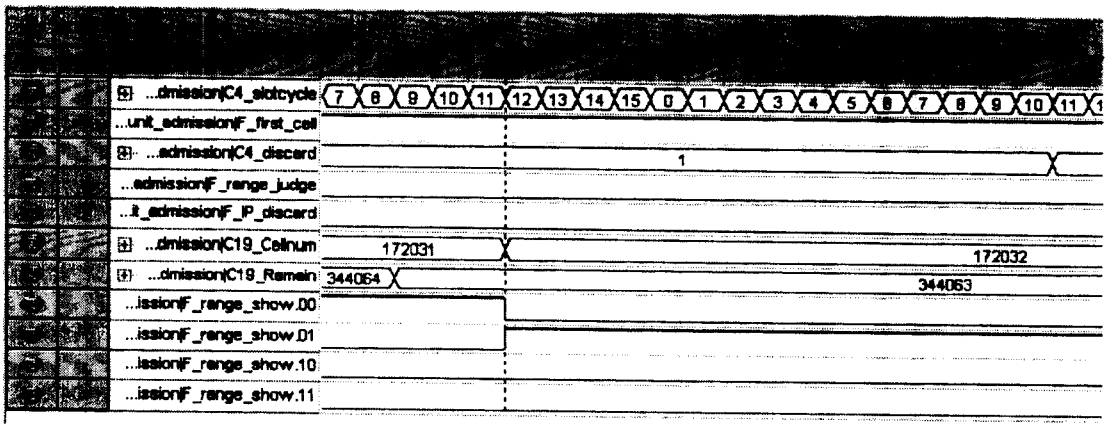

...dmission\C4_slotcycle
...unit_admission\F_first_cell
...admission\C4_discard
...admission\F_range_judge
...it_admission\F_IP_discard
...dmission\C19_Cellnum
...dmission\C19_Remain
...ission\F_range_show.00
...ission\F_range_show.01
...ission\F_range_show.10
...ission\F_range_show.11
172031
172032
344064
344063

图 12

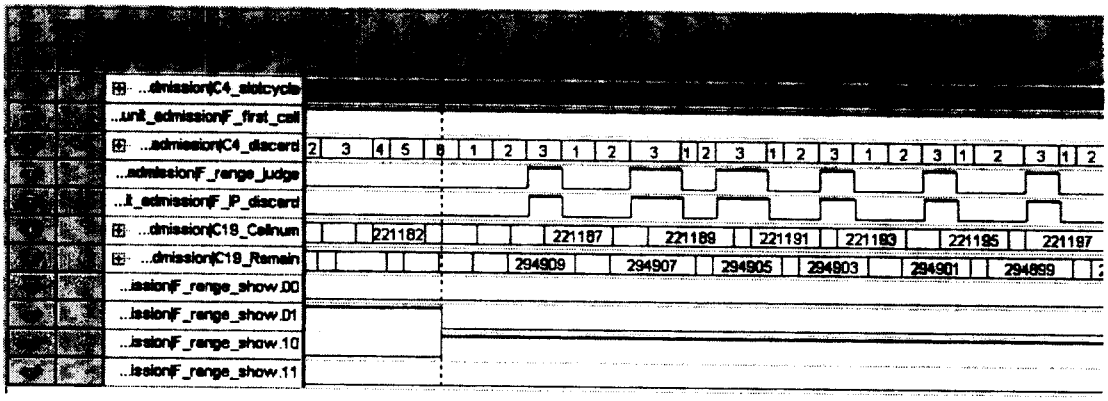

...dmission\C4_slotcycle
...unit_admission\F_first_cell
...admission\C4_discard
...admission\F_range_judge
...it_admission\F_IP_discard
...dmission\C19_Cellnum
...dmission\C19_Remain
...ission\F_range_show.00
...ission\F_range_show.01
...ission\F_range_show.10
...ission\F_range_show.11
221182
221187
221189
221191
221193
221195
221197
294909
294907
294905
294903
294901
294899

图 13

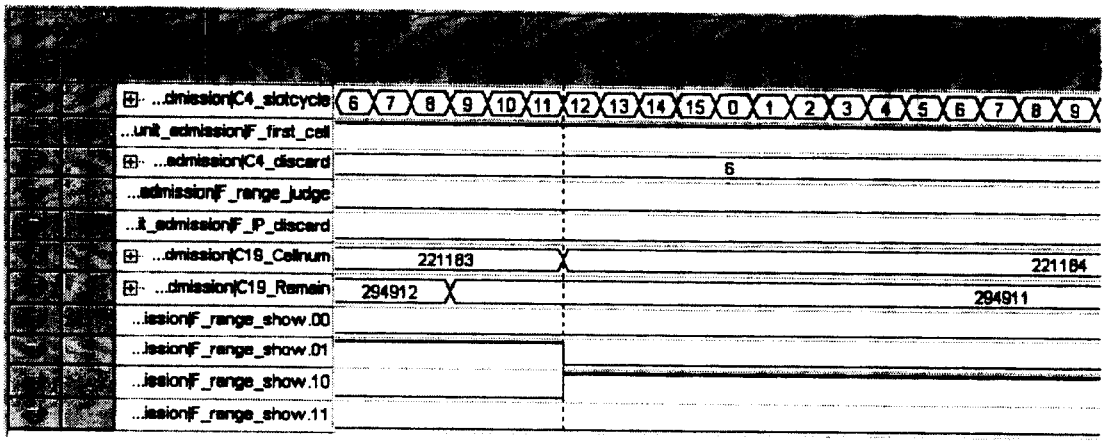

...dmission\C4_slotcycle
...unit_admission\F_first_cell
...admission\C4_discard
...admission\F_range_judge
...it_admission\F_IP_discard
...dmission\C19_Cellnum
...dmission\C19_Remain
...ission\F_range_show.00
...ission\F_range_show.01
...ission\F_range_show.10
...ission\F_range_show.11
221183
221184
294912
294911

图 14

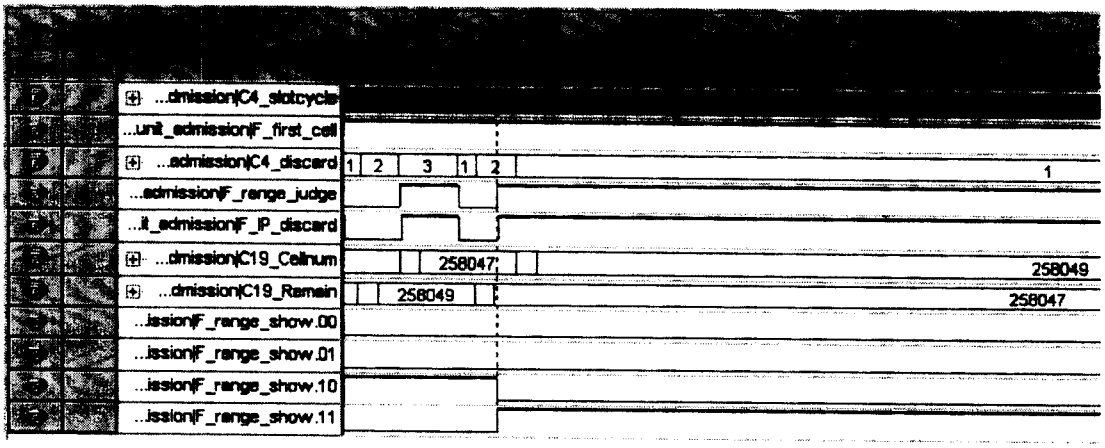

...dmission\C4_slotcycle
...unit_admission\F_first_cell
...admission\C4_discard
...admission\F_range_judge
...it_admission\F_IP_discard
...dmission\C19_Cellnum
...dmission\C19_Remain
...ission\F_range_show.00
...ission\F_range_show.01
...ission\F_range_show.10
...ission\F_range_show.11
1 2 3 1 2
1
258047
258049
258049
258047

图 15

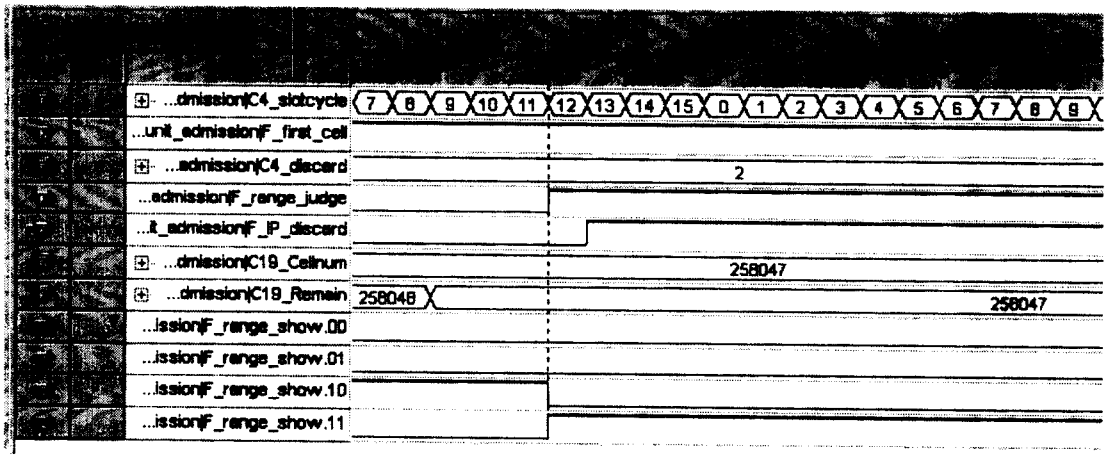

...dmission\C4_slotcycle
...unit_admission\F_first_cell
...admission\C4_discard
...admission\F_range_judge
...it_admission\F_IP_discard
...dmission\C19_Cellnum
...dmission\C19_Remain
...ission\F_range_show.00
...ission\F_range_show.01
...ission\F_range_show.10
...ission\F_range_show.11
7 8 9 10 11 12 13 14 15 0 1 2 3 4 5 6 7 8 9
2
258047
258048
258047

图 16