1. 数据结构

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

RQ的主要内容

body

vqhead

rq 的类型由body的类型决定， body和vqhead 是通过下表一一对应的两个数组。

body 是void\*类型， 用来存放实际的数据

vqhead 的是用来将body由物理内存上的数组组织成逻辑上的链表的数据结构。

vqhead的数据结构如下：

{

int prev;

int next;

vq\_entry\*vq;

}

vqhead 通过prev和next 的形式组织起每个虚拟队列vq.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Prev 0  Next 3 |  | Prev1  Next7 |  |  |  | Prev7  Next10 |  |  | Prev7  Next |

每个rq对应的多个vq, 如上图， 同一个vq的组织形式如上图。Vq中的每一个数据项都通过vq\_entry\*vq; 指针指向他所属的vq。

Rq初始分配的时候所有内容都属于global\_free这个虚拟队列

VQ的主要内容：

vq 的主要数据结构是 vq\_entry.

vq\_entry｛

Int head;

Int tail;

Real\_queue \*rq;

｝

head指向vq第一个vqhead的下标。

Rq 指向vq对应的rq。

1. 使用方法

1.第一步先确定rq的类型， 为rq中的body分配空间。

2.调用init\_real\_queue 函数初始化rq。(GPR系统中主要有login, result和task三种队列， 可参见real\_queue.cpp)

3.用init\_vq\_entry初始化临时虚拟队列。

4.用alloc\_vq\_temp为临时虚拟队列分配空间。

5. get\_body通过索引获取body进行赋值。

6.最后用move\_from\_temp分配的临时队列中的内容复制到自己想要的队列中。（具体流程细节可参照udp.cpp）

3.说明：

get\_head 和 get\_vq\_head的区别是， 前者通过下标索引获取实体队列中的vqhead, get\_vq\_head 则是获取指定的虚拟队列的第一个节点的下标。

每次都要对为队列添加新的内容都要通过临时队列的目的是防止死锁的发生。