ib_core驱动分析

2022年9月25日 14:26

关于 linux workqueue的介绍

https://blog.csdn.net/u012294613/article/details/126666520

主要包括：workqueue、workerpool、poolworker，分别代表任务队列，处理线程队列，和 关联任务队列 + 处理线程队列的 poolworker，然后分别都有结构体定义，以及workqueue的状态机定义，workqueue的处理场景一般用于中断上下文处理，迅速回复，然后从 workqueue中取出信息完成中断下半部的处理。

关于 linux class的介绍

https://blog.csdn.net/kunkliu/article/details/102463149

主要讲解了 class_create 和 class_register函数，对应的是destroy和 deregister函数，里面说的很清楚，class_create主要是在 /sys/class下创建对应的class，device_create是在 /dev 目录下创建对应的 dev设备节点；ib_core.ko中 调用 class_create，就是在 /sys/class/ 下 创建 infiniband文件夹

关于linux netlink的介绍

https://zhuanlan.zhihu.com/p/452045969

Netlink 是一种 内核主动向用户态发起通信的协议，用户态同内核态通信的方式采用系统调用，内核态主动同用户态进行通信采用netlink的方式；至于中断，那是内核态同硬件通信的方式；寄存器则是内核同硬件通信的方式；

ib_core 里面调用 netlink_kernel_create 创建了 netlink socket，其中的recv_msg表示收到的是用户态消息，而不是网卡消息；

rdma_nl_init函数

这个函数定义了 netlink的recv函数，recv函数的具体逻辑取决于cb函数，cb函数来自于一个数组，数组访问index获取具体的回调，应该是取决于用户态传下来的具体消息决定内核态的处理

关于 register_netevent_notifier

https://blog.csdn.net/caihaitao2000/article/details/81604310

介绍了 linux的通知链，register_netevent_notifier是对linux提供的基础通知链的封装，即发生网路事件时，调用 register_netevent_notifier注册的 callback函数

ib_core.ko 里面关注 event NETEVENT_NEIGH_UPDATE

关于 linux neighbor的介绍

https://abcdxyzk.github.io/blog/2019/06/05/arp/

neighbor协议对应的是 ipv4的 ARP协议，通过L3获取L2的地址，为什么需要L2的地址，因为交换机是 二层转发，mac-端口存在映射关系表，当然现在3层交换机是直接ip映射到端

口，问题来了3层交换机ip映射为端口，那么源机器还是需要拿到 dst mac，依然需要ARP，那么三层交换机的好处是什么呢？节省了路由表项？
ib_core 里面关注 NEIGH_UPDATE事件，大胆猜测这个应该是响应 ARP的报文，因为需要同外界通信，保存对端的mac

关于 linux completion机制的介绍

https://blog.csdn.net/qq_40629752/article/details/113747137
简单的说，completion机制就是 线程间的一个同步手段，一个线程去唤醒另一个线程让他做啥事，另一个线程调用 waitup等待别人唤醒他
ib_core.ko 里面初始化了一个 completion变量，还初始化了3个 workqueue，还注册了一个 netevent的 notifier，关注 NETEVENT_NEIGH_UPDATE事件

ib_mad_init函数

ib_mad_init 调用了 ib_register_client，smc-r也调用了 ib_register_client，根据 ib_register_client的源码以及注释，ib_register_client的作用是 申请 client的context，以及 client不为空的时候，调用 client的 ADD方法。

rdma_start_port 和 rdma_end_port 根据 device的 is_switch函数返回 start_port 和 end_port，如果非 switch，则start port为0，end_port的话如果为switch则end_port为0，否则为 代码传入的phys_port_cnt

rdma_cap_ib_mad 的意思是 检测 port是否支持 IB management datagram，即是否支持IB 的管理报文

ib_mad_init触发 ib_mad_register的add函数，触发ib_mad_init_device函数
ib_mad_device_init函数的主要作用：遍历 ib设备的所有 port，针对每个端口，首先调用 ib_mad_port_open函数

ib_mad_port_open函数

该函数里面进行了 发现 PKEY，创建QP，创建CQ，然后post recv了一堆wr，用来接收MAD 消息，但是 CQ的使用方式是 notify_cq，意即使用中断，那么什么时候处理 中断事件呢，谁来处理中断事件呢？