协助解决pru问题

一、解决中断造成的数据不同步问题

在应用调用接收数据时进入阻塞状态等待数据，在固件准备好数据放在共享内存时会触发中断，此时驱动会从休眠唤醒并将共享内存中的数据拷贝到用户空间，如果在这个过程固件又产生中断，这个数据拷贝过程就会被打断，因此需要在数据拷贝前关闭该中断，在拷贝完成或失败后恢复该中断，如下面代码所示：

```
    for(pru_n = 0; pru_n < NUMBER_PRU; pru_n ++) {
        disable_irq(irqnum);    /* 关闭该中断 */

        ....


            if ((ret1 = copy_to_user((BuffRx[pru_n]), ((char*)shared_data->buf_rx),
SHARED_MEMORY_SIZE ))){
                printk("copy_to_user err\n");
                enable_irq(irqnum); /* 恢复该中断 */
                return -1;
            }
            enable_irq(irqnum);     /* 恢复该中断 */

            break;
        }
        enable_irq(irqnum);         /* 恢复该中断 */

    }
```

二、分析固件匹配问题

从测试中发现，pru固件和网口并不是一一对应。按如下设置能触发中断并正常收到数据：

- 应用读写pru1固件对应的共享内存
- 驱动中加载启动pru0固件
- 驱动的拷贝数据函数中只拷贝pru1固件对应的共享内存数据给用户
- 接到pru网口1
- 触发的数中断17，是pru0核的中断

从以上现象可知固件确实是加载运行在pru0核，并且驱动中加载的是pruss2_pru0.bin，但是应用读写的是pru1的共享内存，并且接的也是pru1网口，此时正常触发pru0的17中断和正常交互数据，如果应用读写的是pru0的共享内存或者接的是pru0网口则无法正常交互数据。因此很有可能pruss2_pru0.bin并不是对应的pru0核的固件。