优化完善pru字符设备驱动框架

一、优化驱动接收数据时的框架

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、优化驱动的固件相联部分

在之前的驱动中加载启动两个固件，并且在读取数据返回给用户空间也是两个固件的共享内存都找一遍，这样驱动行为不明确，也不好调试，因此可把pru0和pru1剥离开来。按如下设置来修改驱动为pru0相关。

- 首先应用修改为读写pru0核的共享内存
- 然后驱动的加载函数只保留pru0相关的设置，如设备树节点获取、注册中断和只加载运行pru0固件。
- 在驱动的协助函数中释放相应的全局资源和上一步的pru0资源
- 在驱动的RecieveFrame函数中只读pru0核的共享内存数据

同理，pru1相关设置也一样。