WIFI驱动分析

Revision History

Date	Issue	Description	Author
<25/02/2009>	<0.5>	First draft	wylhistory

目录

1. Abstract

.这里主要讲的是我对WIFI驱动的理解。

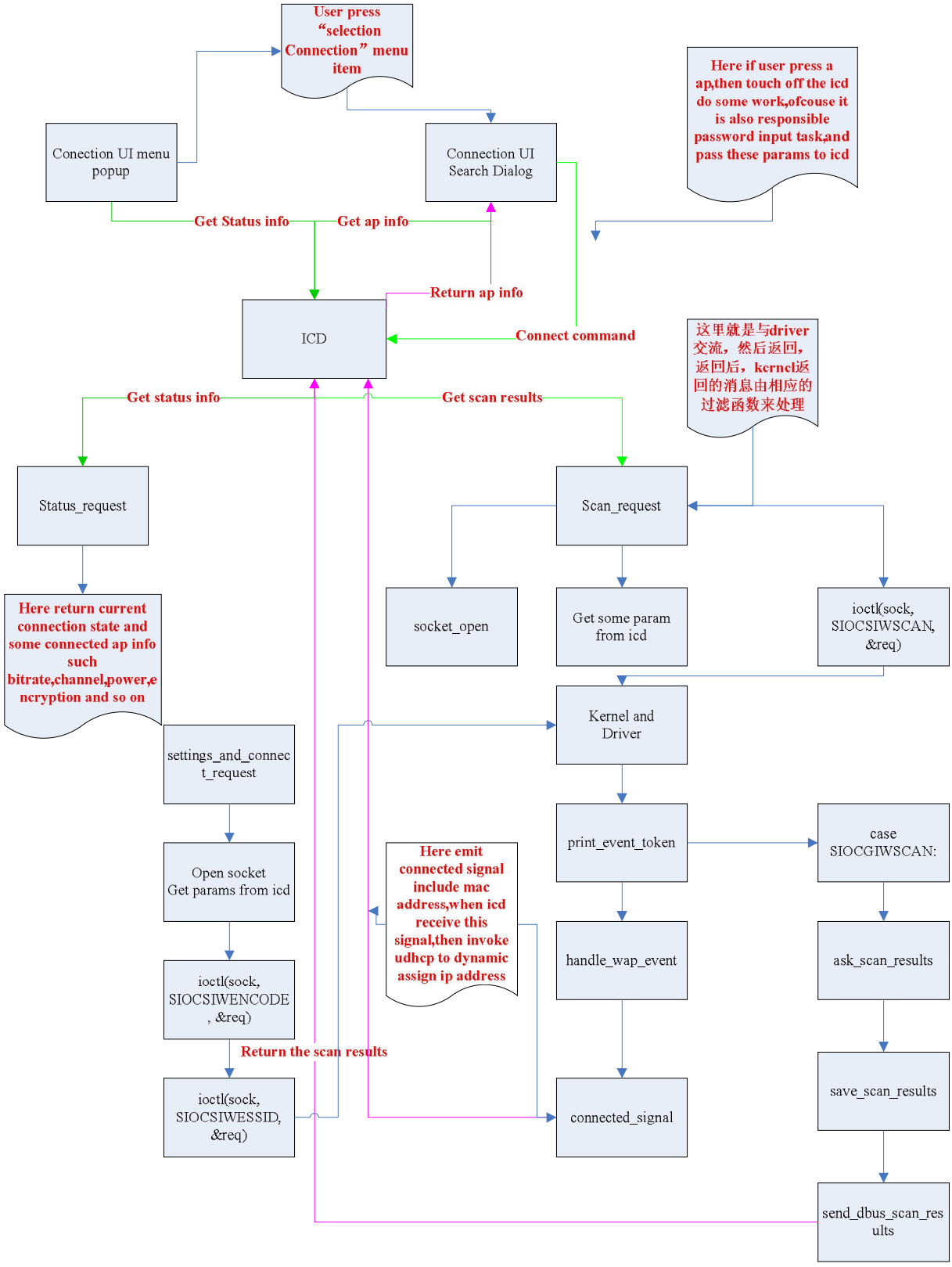
2. Introduction

因为将要负责WIFI驱动，所以就开始了WIFI驱动的学习，主要分析了WIFI驱动的初始化，数据的发送流程以及和电源管理相关的部分。

3. 用户使用流程

通常用户的做法就是打开一个socket，调用一个ioctl，等待消息返回，收到消息后继续做下面的事情，然后又等待内核消息的返回，如此循环。

比如我们的系统的流程就是这样的：



4. WIFI驱动的初始化

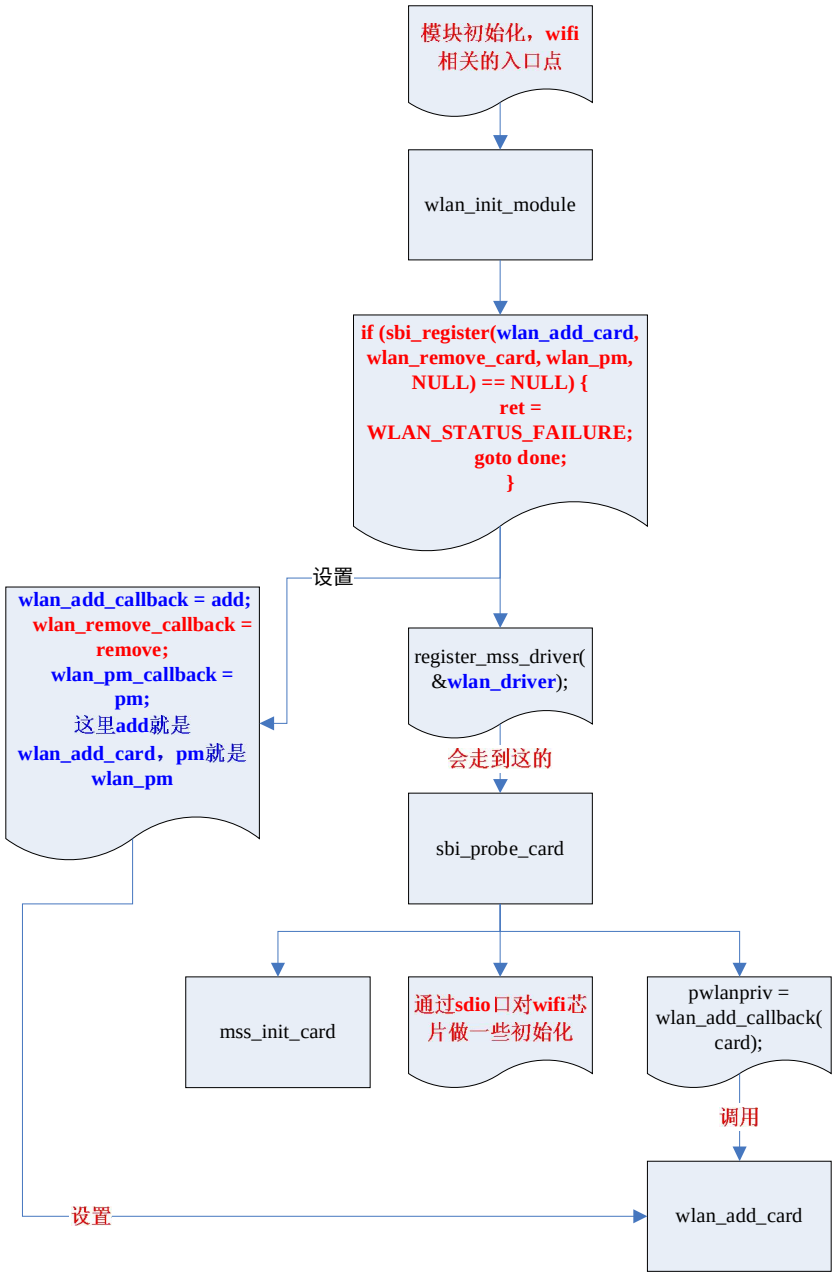


图4.1 wifi模块初始化

这里主要的工作就是注册mmc driver，注册回调函数，包括电源管理的，包括设备添加的，当然还有一些硬件初始化；

这里最重要的是wlan_add_card这个函数做了很多事情，我把我没有分析的代码都略过了:

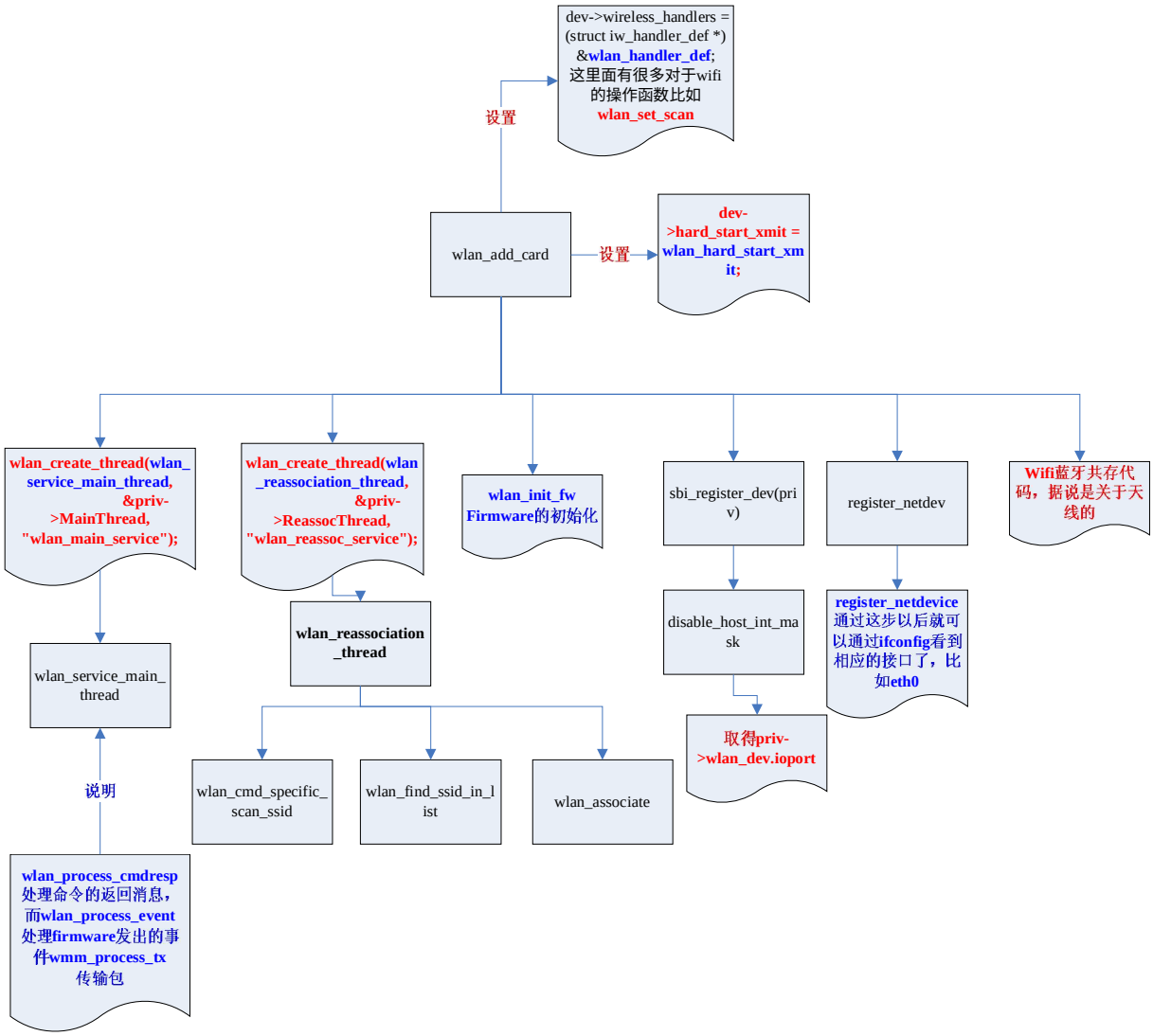


图4.2wlan_add_card的逻辑

这里创建了两个线程，一个用来处理基本的输入输出，那就是wlan_service_main_thread，一个用来负责重新连接AP（当自动断开的时候），那就是wlan_reassociation_thread 还做了一些firmware的初始化；然后就是注册mmc设备，取得端口号；接着就是注册一个网络设备，用来供上层访问，这时候就可以通过ifconfig来看到输出了，比如是eth0；最后是一些用来使蓝牙和wifi能够共存的代码；

5. 数据的发送

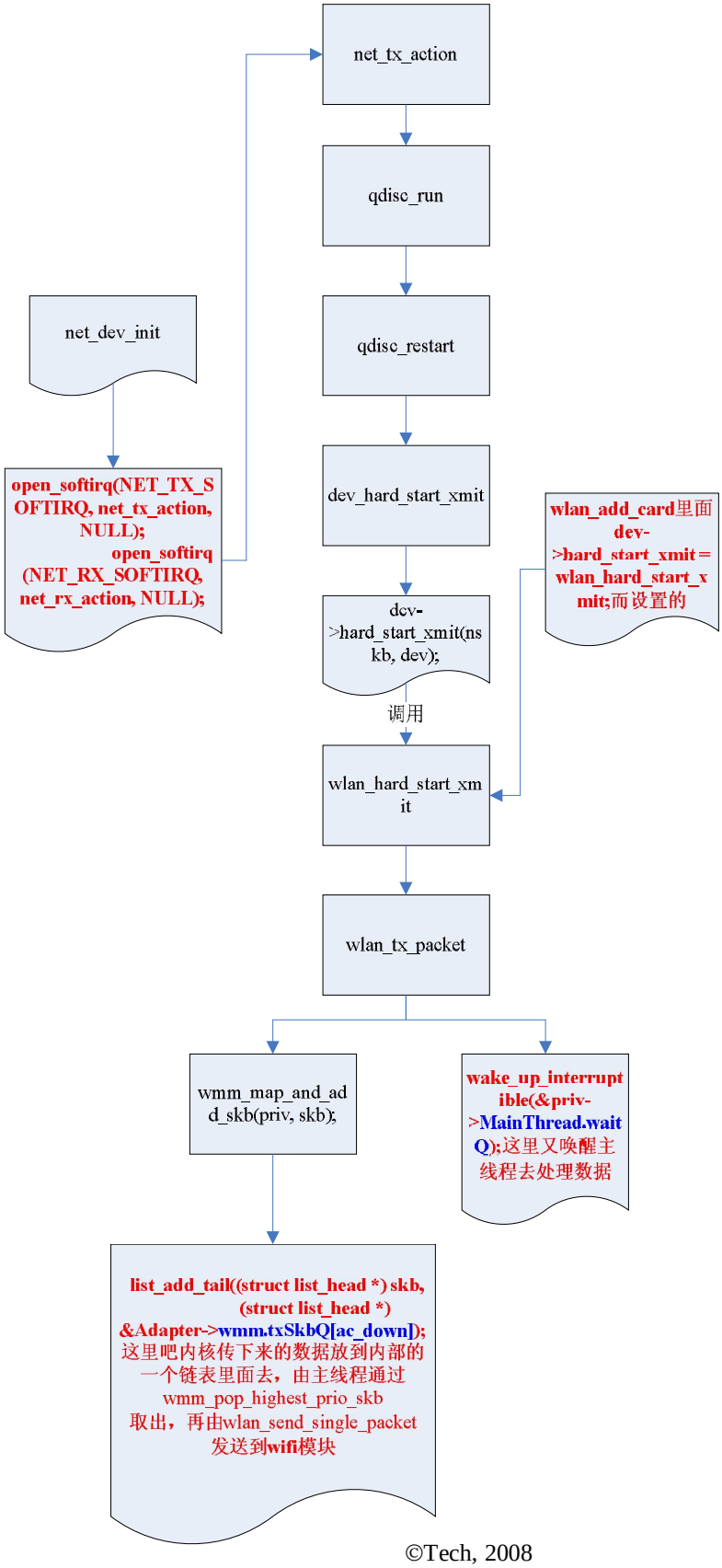
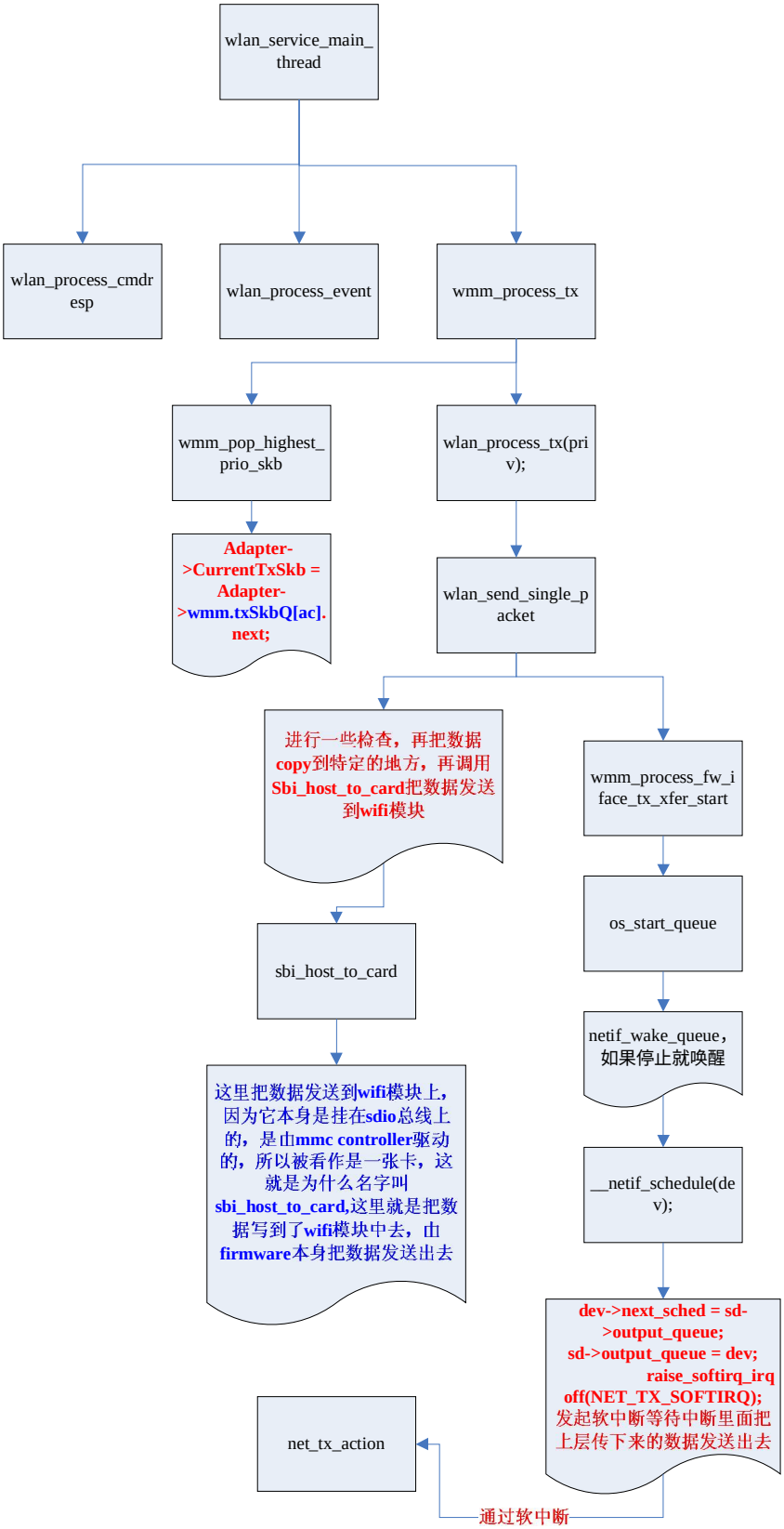


图5.1发送数据的触发

数据的发送请求从tcp/ip层传到了这里，于是通过唤醒WIFI的主线程的处理函数来发送具体的数据请求；

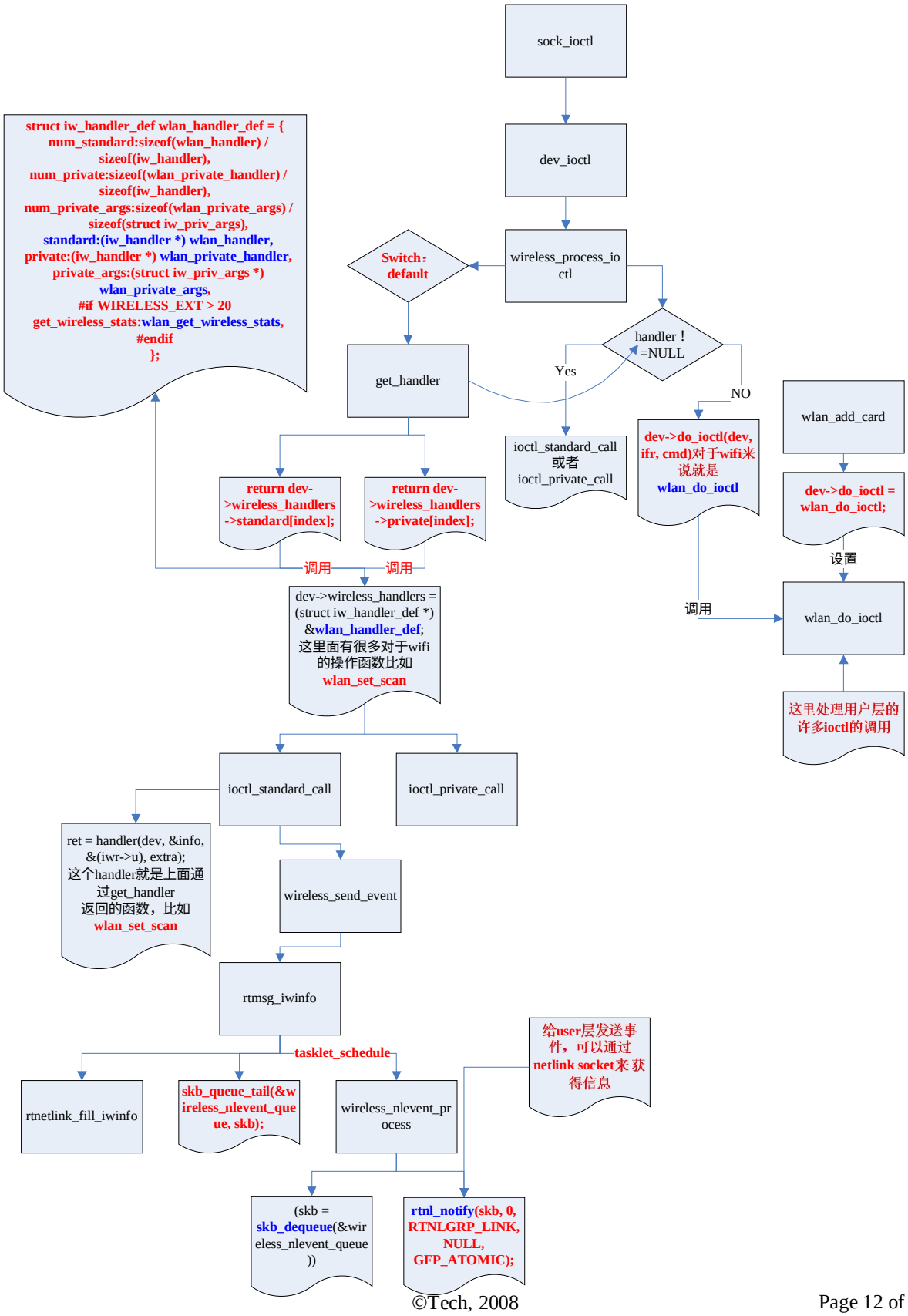
下面看主线程里面的数据发送：



可见最后数据是通过mmc总线发送到了wifi模组了，而且开始调度下一次的数据发送，至此，数据的发送过程已经分析完了，下面是ioctl的调用逻辑

6. ioctl的调用逻辑

之所以要分析这个，是因为上层和WIFI驱动打交道的方式，多半是通过ioctl的方式进行的，所以…看看它的调用逻辑：



可以看到WIFI模块对ioctl的处理非常复杂，主要是要处理许多标准的调用，也要处理一些私有的调用；后面还要通过rtnl_notify给上层用户发送消息，这里也是一套机制，我也就不细说了。

7. 电源管理相关的调用逻辑

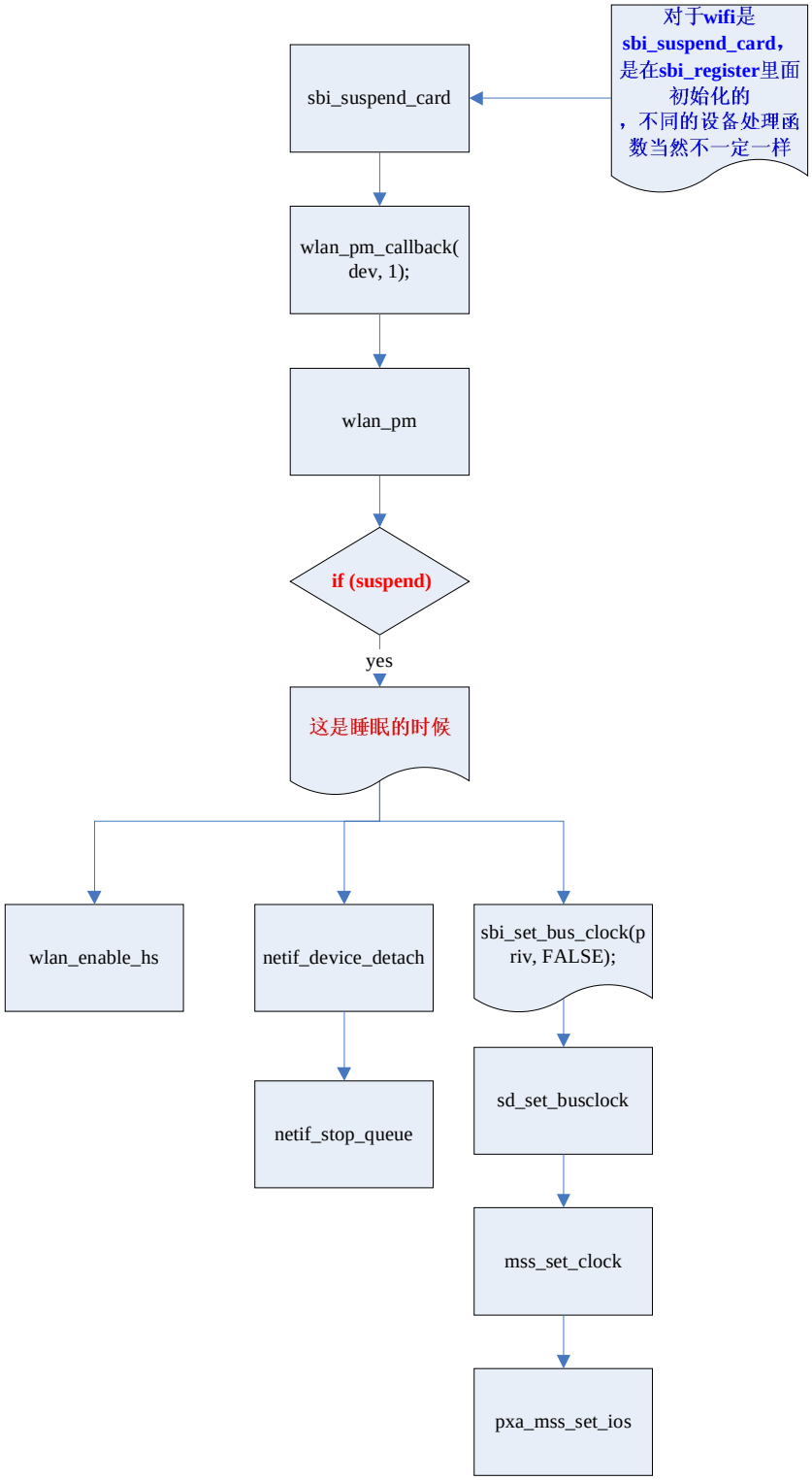


图7.1pm相关的逻辑

在系统要求睡眠的时候会调用到sbi_suspend_card,由此开始给WIFI模块发送相关信息，并且detatch网络设备，停止收发队列的处理，停止mmc总线等等，唤醒过程没有分析；

8. 剩下的问题

1，逻辑似乎不复杂，但是细节实现不清楚，这个可能需要具体问题具体分析了；
2，只分析了发收过程，接收过程类似，但是没有去分析；
3，数据到ip层后没有去分析。
4，不同加密方式的AP的连接，没有分析，将来或许会用到；