

# IEEE802.15.4 的 MAC 协议实现

吴 考

(北京航空航天大学 电子信息工程学院, 北京 100083)

**摘 要:** IEEE802.15.4 协议是无线传感网中多采用的一种低速率、低功耗、低成本的无线通信协议。Imote2 是 Intel 公司推出的高集成度的无线传感器平台。文中介绍了 Imote2 平台以及 IEEE802.15.4 协议, 及网络驱动程序的体系结构, 给出了实现 IEEE802.15.4 的 MAC 协议过程中的关键函数, 还对 MAC 协议实现中的中断服务响应以及时钟等关键问题进行了描述。并分别对节点的工作性能及组网时节点的性能及损耗进行了测试, 给出了系统的性能。结果显示, 该种实现方法有效地保证了系统的性能并且可以达到无线传感网组网需求。

**关键词:** IEEE802.15.4; MAC 协议; Linux 网络驱动

**中图分类号:** TP274+.2

**文献标识码:** A

**文章编号:** 1673-629X(2010)04-0008-04

## Realization of IEEE802.15.4 MAC Protocol

WU Kao

(School of Electronic and Information Engineering, Beihang University, Beijing 100083, China)

**Abstract:** The standard of IEEE802.15.4 protocol is a wireless protocol designed for WPAN with low bit-rate, low energy-consuming and low cost. Imote2 is a high integrative wireless sensor platform designed by Intel. This paper introduces the IEEE802.15.4 protocol and the Imote2 platform, and then talks about the architecture of the Linux network driver. Then this paper discusses the most important functions of this driver and gives the performance and the energy consuming results in simple WSN system. The paper also describes several important things such as interrupt response and clock system. The results show that the realization can reach the requirements of WSN system.

**Key words:** IEEE802.15.4; MAC protocol; Linux network driver

## 0 引 言

IEEE802.15.4 协议<sup>[1]</sup>由于其低功耗、低成本的特性目前广泛应用于无线传感器网络之中, Imote2 为无线传感器网络的搭建和研究提供了很好的平台。文中主要介绍了 IEEE802.15.4 的 MAC 协议在 Imote2 平台上的实现以及 Linux 网络驱动的架构及驱动实现过程中的关键问题, 并给出了实际测试中无线传感器节点的性能及能耗。

## 1 IEEE802.15.4 的 MAC 协议

随着无线技术的发展和人们需求的多样化, WPAN(无线个域网)作为 10 米范围内传输数据的无线技术得到了很多应用并且受到人们的重视, IEEE

802.15.4 协议正是为 WPAN 设计的一种低功耗、低速率、低成本的无线通信协议。IEEE802.15.4 包含物理层和 MAC 层的两个规范, 主要规定了 CSMA-CA 信道访问机制和不同设备 MAC 层之间的可靠传输。CSMA-CA 是带冲突避免的载波多路侦听访问机制, 降低了无线信道传输数据时发生冲突的可能性, 提高了信道传输数据的成功率。文中主要就是在 Imote2 平台上实现了 IEEE802.15.4 的 MAC 协议。

## 2 Imote2 平台介绍

Imote2 是 Intel 公司推出的一款专门用于无线传感器网络应用的高级传感器网络平台, 采用了功能强大并且功耗极低的 XScale 处理器, 并集成了 2.4GHz 天线和 CC2420 无线模块<sup>[2]</sup>。它提供了多种操作系统的选择, 包括 TinyOS 和 Linux, 文中采用了扩展性更好, 功能更为丰富的 Linux 操作系统。

Imote2 平台虽然集成了 CC2420 无线模块, 但是自带的驱动程序并未将其映射成为一个网络设备来使用, 因此, 文中旨在通过新的驱动程序模块将 CC2420

收稿日期: 2009-07-15; 修回日期: 2009-10-30

基金项目: 国家自然科学基金(60672103)

作者简介: 吴 考(1985-), 男, 安徽芜湖人, 硕士研究生, 研究方向为无线传感网、嵌入式; 导师: 杨晨阳, 教授, 研究方向为 MIMO、UWB 等。

映射成为一个标准的网络设备,并实现 IEEE802.15.4 的 MAC 协议中无信标(Non-Beacon)模式下的 CSMA-SA 协议,为利用 Imote2 平台进行无线传感器网络的研究提供可能。

### 3 MAC 协议的实现

MAC 协议的实现是通过 CC2420 的驱动程序来实现的,下面分析了网络驱动程序的体系结构并介绍了本驱动的主要结构及重要函数。

#### 3.1 Linux 网络驱动程序的体系结构

网络驱动程序的体系结构如图 1 所示<sup>[3]</sup>,它由四层组成,其中每一个设备都被映射为一个设备接口,网络协议接口层将发送数据包的命令下发到网络设备接口层,即 net\_device 结构,再由驱动的功能层负责将数据包发送到硬件,最终经由设备物理媒介层发出包。而接收的过程正好相反,设备物理媒介层接收到有效的数据流后会发送给设备驱动功能层处理,再经由网络设备接口层传递给上层协议接口,供上层网络协议使用<sup>[4]</sup>。由于 Linux 内核源代码自带了设备物理媒介层的功能,因此需要完成的网络驱动程序主要是设备驱动功能层。

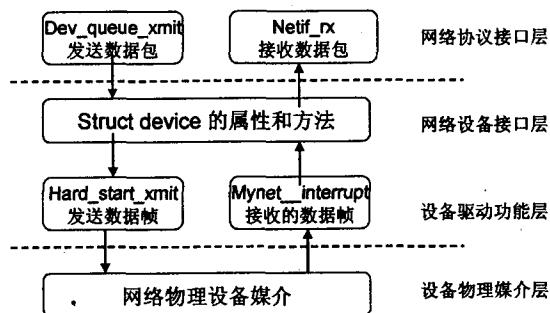


图 1 Linux 网络设备驱动程序的体系结构

#### 3.2 IOCTL 的设定与实现

##### 3.2.1 IOCTL 的调用

IOCTL 是设备驱动程序中对设备的 I/O 通道进行管理的函数,实现对设备的特性(如 CC2420 的 MAC 地址、传输信道、混杂模式等等)的控制<sup>[5]</sup>。在应用层中 Ioctl 的调用方式如下:

```
int ioctl(int fd, ind cmd, ...);
```

其中,fd 是用户程序打开设备时使用 open 函数返回的文件标示符(对于 CC2420 来说为 wpan0),cmd 是用户程序对设备的控制命令,省略号是补充参数。文中给 IOCTL 提供了丰富的参数,包括所有控制命令和性能统计量来方便上层对 CC2420 进行控制或者提取相应的统计量来判断系统性能。

##### 3.2.2 IOCTL 的通用接口

IOCTL 的通用接口包括 PIB 以及 MIB 信息的读

取以及设定。对 PIB 信息的读取:应用层调用 ioctl 的方法为 int ioctl(int fd, ind cmd, ...);读取 phy-PIB 信息的 cmd 关键字为 IOCTL\_GET\_PIB,读取 mac-PIB 的关键字为 IOCTL\_GET\_MIB。用于在驱动和应用层传输数据的是内核中定义的 ifreq 型结构里的 ifr\_ifru ifru\_data 成员,需要先定义一个与 ieee802154\_phy-PIB 结构相同的结构,将指向它的指针强制转换为一个空指针,赋给 ifr\_ifru ifru\_data 作为输入。驱动层接收到相关指令后所做的操作就是将驱动一直维护的 ieee802154\_phy-PIB 结构的指针赋给 ifr\_ifru ifru\_data。因此,上层可以通过这个指针得到所需要查询的各个 phy-PIB 值。

对统计量信息的读取与 PIB 信息的读取类似,关键字为 IOCTL\_GET\_STATISTIC。

对 PIB 信息的设定:除了一些标准规定的定值外,其余 PIB 变量均可设定。设定方式分为 insmod 命令设定参数的方法,以及利用 IOCTL 接口设定的方式。

#### 3.3 MAC 协议过程的实现

##### 3.3.1 CSMA-SA 状态机

MAC 协议的过程主要是通过以下这个状态机来实现的:

- 1) 引入 CSMA/CA 协议状态机(见文献[1]中图 3,如文中图 2 所示):IDLE、BACKOFF、WAITACK;
- 2) 结点初始状态为 IDLE,如果有包要发送则进入 BACKOFF 状态,成功发送非广播包后设置等待 ACK 最大时间定时器,进入 WAITACK 状态;
- 3) 如果结点等待 ACK 超时,但包重传次数未超过标准规定的最大值,则重新发送同一包,进入 BACKOFF 状态;如果结点等待 ACK 超时,且包重传次数超过标准规定最大值,则此包发送终止,进入 IDLE 状态;
- 4) 如果结点在等待 ACK 的最大时间内收到正确的 ACK,则此包发送过程结束,进入 IDLE 状态(见文献[1]中图 6)。

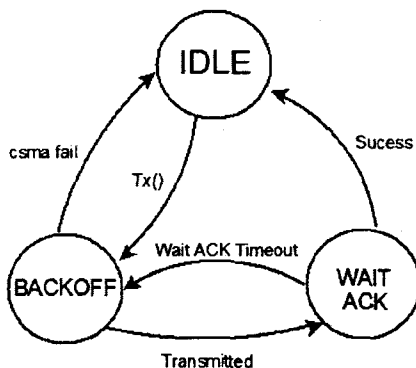


图 2 非时隙 CSMA/CA 协议状态机

### 3.3.2 具体实现函数

●设备的初始化函数 `init()`: 它将收到一个指向正被初始化的设备的指针, 并且将 `dev` 结构填充<sup>[6]</sup>。

●发送函数 `cc2420_tx()`: 它在上层协议有数据需要传输时被调用, 该函数被调用时, CSMA-CA 状态机处于空闲 (IDLE) 状态, 在被调用后, 它将完成以下操作:

- a. 初始化 CSMA-CA 变量;
- b. 产生随机退避时间并设置定时中断;
- c. 将待传输数据写入 CC2420 发送缓存;
- d. CSMA-CA 状态机由空闲进入退避 (BACK-OFF) 状态, 发送数据的过程将在定时中断响应函数中进行。

●定时中断响应函数 `cc2420_timeout()`: 当定时中断到达后, 定时中断响应函数将被调用, 该函数实现了非时隙 CSMA-CA 协议的随机退避过程和 ACK 帧等待过程。该函数里完成了以下操作:

- a. 查询 CC2420 的 CCA 管脚, 检测当前信道是否空闲。
- b. 若信道空闲则开始传输并设置等待时间并使得状态机进入 ACK 帧状态。
- c. 若信道忙, 则重新重传计数器, 如果重传次数超过了最大值, 则清楚发送缓冲区内的数据, 状态机回到空闲。

●接收中断响应函数 `cc2420_rx()`: 当 CC2420 接到有效数据后, 其 SFD 管脚会产生跳变, 并作为一个中断信号被捕捉并进入接收中断响应函数。该函数将从 CC2420 接收缓存区中读取接收数据, 并根据 CSMA-CA 协议状态机当前状态进行处理:

若当前 CSMA/CA 状态机处于等待 ACK 帧状态, 且接收到的是有效的 ACK 帧, 则需要取消等待 ACK 帧的定时中断, 表示已经正确地接收到了 ACK 帧。同时, CSMA/CA 状态机将切换到空闲状态, 并打开上层传输队列, 准备传输新的数据帧。

若当前 CSMA/CA 状态机处于空闲状态, 且接收到的是有效数据帧, 则应该过调用内核函数 `cc2420_rx()` 将该数据帧传递给上层协议。因为 CC2420 模块提供了自动 ACK 功能, 所以驱动不必对此进行任何操作。

●接收函数 `cc2420_rx()`: 首先检查接收到得数据包是否符合能量门限的要求, 如果能量不够则丢弃, 如能量达到门限则将接收到的包加入队列, 同时依据目前的状态机状态来确定如何处理收到的数据包。

### 3.4 几个关键问题

在实现 IEEE802.15.4 的 MAC 层协议过程中, 由

于 Linux 系统, 以及 Imote2 平台的特殊性, 使得在一些问题的处理上会给实际性能带来很大影响。以下就是在网络设备驱动程序的实现过程中影响实际性能的关键问题。

#### 3.4.1 时钟问题以及对性能的影响

Linux 系统的时钟周期是 10ms, 而 802.15.4 的 MAC 协议里定义的随机退避时间都是 ns 量级的, 因而在驱动中涉及退避时如果利用系统的时钟会严重影响系统的性能<sup>[7]</sup>。实际操作中, 选择了直接调用 PXA271 的时钟, 精确度达到了 ns 量级。第四部分的 PING 测试结果可以反映直接调用硬件时钟给性能带来的提升。

#### 3.4.2 中断响应问题

设备通过中断来告知软件可以对它进行操作, Linux 为中断处理提供了很好的接口。中断分为硬中断和软中断两种<sup>[8]</sup>。硬中断是指强行独占系统内存等待, 软中断是指系统只是等待中断信号的到来, 这段时间内仍旧可以进行其他响应。一般来说都会选用软中断, 因为频繁的硬中断会极大地降低系统性能, 但是, 由于 802.15.4 协议的特殊性, 由于其受到碰撞的概率很大, Linux 系统时隙又比较大 (10ms), 因而经常会造成丢中断的后果, 而在状态机中丢中断就意味着一定要进入循环等待直至超过最大重试次数而放弃, 这在节点数较多的情况下尤为明显, 因此实际操作中, 在独占内核和协议效率间进行了折衷考虑。当随机退避时间较短时 (2ms 以下), 驱动不释放内核资源, 独占内核进行随机退避, 以换取高效率。当退避时间较长时, 驱动设置定时器中断, 并释放内核, 通过 PXA271 定时器中断完成退避, 以减少对内核响应时间的影响。实际测试发现, 这样做有效地提升了系统性能, 并降低了碰撞的几率。

## 4 实际性能测试

为了对所实现的 MAC 协议进行测试, 将 CC2420 的驱动导入到 Imote2 节点中, 并且进行 PING 测试、吞吐量测试以及无线传感网组网性能测试。

### 4.1 PING 测试

PING 测试时通过发送一个 ICMP 回声请求消息给目的地并报告是否收到所希望的 ICMP 回声应答, 用以检测网络是否通畅以及传输速率快慢。测试结果如表 1 所示, 第一行是利用 PXA271 的时钟时的返回时间, 第二行是利用 Linux 系统时钟时的返回时间。可以看出, 最终的结果在 10ms 左右浮动, 并且长时间 (半小时以上) 运行稳定无误。

表1 PING 测试结果

1/ms	2/ms	3/ms	4/ms	5/ms
23.4	10.0	13.2	11.3	9.7
113.2	89.3	84.3	90.3	70.3

## 4.2 吞吐量测试

吞吐量是关于网络容量以及网络运载能力的重要参数,这里采用了 Linux 下的 nttcp 软件分别测试了在 TCP 和 UDP 协议下系统的吞吐量。多次测试平均的结果为 TCP 下 13.2kB,UDP 下 15.3kB,达到了 802.15.4 协议理论值 250kb 的一半左右。差距来源于硬件响应的时间损耗以及 TCP、UDP 层的反应时间。总体上符合应用需求。为了测定具体传输数据时的性能,采用了两个节点之间传输文件的方式,其传输速率也基本稳定在 13kB 左右。

## 4.3 组网性能测试

为了测试该协议在实际无线传感网中的性能,搭建了简单的无线传感器网络。网络层采用 raw-socket 传输,组网规则是简单的分层广播式组网。组网结果如图 3 所示,图中列出了 10 个节点及某次组网完成后的逻辑关系图。同一个大圆圈内代表相邻,箭头代表继承关系。其中 1 号节点为根节点。

通过对组网过程中搜集的发送及接收数据估算了网络建立过程中驱动的实际性能及其与理论值的对比。测试结果表明,每次组网平均节点收发字节数为 120Byte,在节点数分别为 10 个和 20 个的情况下,组网过程均在瞬间完成。测试结果表明该协议实现后在实际无线传感网中的性能达到了要求。

## 5 结束语

网络驱动程序的开发和一般的应用软件开发有很大的不同,因为直接同硬件打交道,所以一点小问题都很容易造成整个系统的崩溃,并且驱动程序的效率高

低直接影响到协议效率的发挥。

文中所实现的 IEEE802.15.4 的 MAC 协议,在真实工作环境中达到了较好的性能水平,并且可以在进一步对无线传感器网络的研究中得到广泛应用。

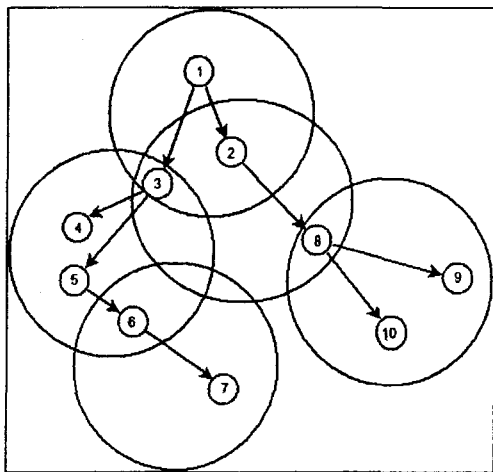


图3 无线传感器网络组网示意图

## 参考文献:

- [1] IEEE802.15 Group. IEEE Standard 802.15.4[S]. 2003.
- [2] Intel 公司. Imote2 Data Sheet[R]. US: Intel, 2005.
- [3] 俞坤师,刘有源,曹小华,等.基于 CC2420 的无线传感节点平台研究[J].嵌入式系统应用,2008(2):17-19.
- [4] 赵 洁,丁香乾.嵌入式 Linux 网络驱动程序的开发及实现原理[J].嵌入式系统应用,2008(2):64-66.
- [5] Corbet J, Rubini A. Linux Device Driver[M]. 3rd ed. [s.l.]: O'Reilly, 2006.
- [6] 刘 森. 嵌入式系统接口设计与 Linux 驱动程序开发[M]. 北京:北京航空航天大学出版社, 2006.
- [7] 李善平,刘文峰,李程远. Linux 内核源代码分析大全[M]. 北京:机械工业出版社, 2002.
- [8] 卞海舟,方 晨,胡 晨. 802.15.4 无线个域网网络协调器的驱动设计[J]. 电子器件, 2008(2):695-697.

(上接第7页)

- [4] 张文修,吴志伟,梁吉业,等. 粗糙集理论与方法[M]. 北京:科学出版社, 2001:1-10.
- [5] 王国胤. Rough 集理论与知识获取[M]. 西安:西安交通大学出版社, 2001.
- [6] 覃政仁. 基于 Rough + Set 的海量数据挖掘算法研究[D]. 重庆:重庆邮电学院, 2004.
- [7] 姚辉学,卢章平. 海量数据多边形布尔运算的区域分割算法[J]. 中国图象图形学报, 2007, 12(3):552-557.
- [8] 伍 东,李 建,税 敏. 海量数据并行压缩算法研究[J]. 山西电子技术, 2007(2):85-87.
- [9] 孔德松. 对多维数据存储技术的研究[D]. 武汉:武汉理工大学, 2006.
- [10] An A J, Shan N, Chan C, et al. Discovering Rules for Water Demand Prediction: An Enhanced Rough-set Approach[J]. Artificial Intelligence, 1996, 9(6):645-653.
- [11] Wu X D, Zhang S C. Synthesizing High-Frequency Rules from Different Data Sources[J]. IEEE Transaction on Knowledge and Data Engineering, 2003, 15(2):353-367.
- [12] 张文修,姚一豫,梁 怡. 粗糙集与概念格[M]. 西安:西安交通大学出版社, 2006:30-34.