# 面向高速工业无线网络的 TDMA MAC 协议设计与实现

林俊如[1,2] 曾 鹏[1] 于海斌[1]
(中国科学院沈阳自动化研究所工业信息学重点实验室 沈阳 110016)[1]
(中国科学院研究生院 北京 100039)[2]

摘 要 随着无线技术的发展,在工业自动化中引入无线技术已经成为一种潮流,同时 TDMA 机制由于可以完全避免冲突,因此在大规模高实时要求的面向车间的无线技术研究中受到了广泛关注,但现在面向车间的高速实时无线技术的研究大都局限于软件仿真。通过在商用 802.11 硬件上设计并实现面向车间级工业应用的高速工业无线 TDMA MAC 层协议,建立了一个基于 TDMA 的面向工业应用的高速无线网络原型系统,实验结果表明实验平台完全能够支撑毫秒级时隙调度的 TDMA 应用。

关键词 工业无线网络,802.11,TDMA
中图法分类号 TP393 文献标识码 A

## High Data Rate Wireless Industrial Networks Test-bed for TDMA MAC Development: Design and Experimentation

LIN Jun-ru[1,2] ZENG Peng[1] YU Hai-bin[1]
(Key Laboratory of Industrial Informatics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China)[1]
(Graduate School of the Chinese Academy of Sciences, Beijing 100039, China)[2]

**Abstract** With the development of wireless technology, using wireless technology in industrial environment becomes more and more popular, meanwhile researchers pay more attention to TDMA mechanism due to its capability to avoid collision which makes wireless communication behavior much more determinate, but most current researches are limited to simulations and experimental studies are rare. To bridge this gap, we developed and implemented a high data rate wireless networks testbed based on TDMA mechanism and commodity IEEE 802.11 hardware, the test-bed can be used for filed level industrial wireless technology's development and experiment, and experimental evaluation shows the test-bed can support TDMA applications that require time slot schedule in millisecond.

**Keywords** Wireless industrial networks, 802.11, TDMA

## 1 引言

继低速的过程自动化之后,无线网络技术已经开始在高速工厂自动化领域应用,并逐步成为发展的热点。在面向车间级工厂自动化中的一般有线替代、机器人末端执行器、轨道挂载设备、旋转设备及移动资产管理方面,无线技术具有易安装、易维护的优势,并且能够避免设备因移动导致的线缆易老化,滑环之间电力接触易失败等问题,因此在面向高速的车间级的工厂自动化中无线技术非常具有应用潜力。高速的工厂自动化要求处理事物速度快及控制精度高,并且要求通信系统能够满足:在可靠性为 99.99999% 的要求下离散信号的确定性响应时间在 10ms 至 100ms 之间,同时网络规模能够支持从百点至千点的节点个数。在大部分情况下,系统的数据交换的总量不大但由于其严格的实时性要求,导致需要系统的传输速率在 Mbit/s 的量级[1],因此 IEEE 802.11 成为了无线高速工厂自动化通信主要的候选技术。

TDMA 技术是实现工厂自动化应用高可靠、高实时的重要选择方案。在工厂自动化应用中一个非常重要的要求为通信时间存在一个合理的上限,但现在 IEEE 802.11 等无线技术普遍采用的 CSMA 机制因其随机冲突化解算法会导致通信延时的不确定性,而工厂自动化中较大的网络规模和节点密度容易引起节点进行退避操作,因此在工厂自动化应用中不宜直接采用 CSMA 机制。TDMA 机制通过时间片的方式调度整个系统的通信,它能够保证在一个时间片内不存在多个节点竞争信道的情况,能够满足工厂自动化对时延确定性要求,是实现工厂自动化应用高可靠、高实时的重要选择方案。

由于价格的低廉和广泛的应用,802.11 设备已经成了开发和评估新的无线系统和应用的事实上的选择,同时由于 Atheros 公司开放了其 802.11 射频芯片在 Linux 下的驱动程

本文受国家自然科学基金(60704046, 60725312, 60804067),国家科技重大专项基金资助项目(2010ZX03006-005-01)资助。
林俊如(1986- ),男,博士生,主要研究方向为无线传感器网络、嵌入式系统等,E-mail: linjunru@sia.cn;曾 鹏(1976- ),男,博士,研究员,博士生导师,主要研究方向为无线传感器网络、工业无线传感器网络等;于海斌(1964- ),男,博士,研究员,博士生导师,主要研究方向为现场总线、工业无线通信、传感器网络及嵌入式系统等。

序源码(MadWiFi, ath5k),因此现有的很多工作都是在 Linux 操作系统和 Atheros 芯片上展开的。但是现有的大部分工作特别是在 TDMA 机制实现方面都面向非工业领域,并没有考虑到工业领域中的一些应用特点。例如: 1) 在 Linux 系统中 TCP/UDP 等高层协议不适合于现场级设备的通信,但用户层程序往往无法直接调用处于内核态的链路层功能,而现场级设备的通信协议中很多都只包含物理层、链路层和应用层。2) 现有的商业硬件往往都针对 CSMA 机制做了相应的优化,数据发送的时机存在较大的随机性难于精确掌握,而对发送时机的控制是 TDMA 协议实现的基础。3) 工业应用中报文的长度非常小,为了减少协议的通信开销,需要重新定义帧格式,但商业硬件往往根据 IEEE 802.11 协议的同步机制有相应的优化,采用新的帧格式后可能无法利用到硬件特性,完全依靠软件进行同步会导致同步精度很低。

因此在开发基于 Linux 和商用芯片实现面向工厂自动化的高速工业无线 TDMA 网络原型系统时做了以下工作:

1. 以 Atheros 在 Linux 上的驱动程序 ath5k 和 Netlink 技术为基础;提出了可灵活配置的通信协议编程架构,我们利用 ath5k 能够灵活控制帧结构从而为编写通信协议提供了很好的便利;同时利用 Netlink 技术与用户层进行交互,使用户能够在应用层快速开发测试、仿真等应用程序。

2. 以 ath5k 为基础,我们实现了对数据发送时机较为精确的控制,实现了一个简单的基于 TDMA 的 MAC 协议,同时针对工业应用中的特点及实验中的问题提出了相应的解决方案。

3. 提出了在自定义的帧格式的情况下如何充分利用硬件特性的同步算法,实现了在 3.2s 同步周期的情况下同步误差低于 20us 的同步精度。

本文第 2 节绍了在商用硬件上实现 TDMA 协议的相关工作,同时也介绍了在面向工厂自动化的无线技术在链路层实时性方面的一些工作;第 3 节提供了一个在 Linux 上工业协议开发架构;第 4 节提供实现 TDMA 功能及时间同步的细节;第 5 节给出实验验证工作并对实验结果中出现的问题提出了相应的解决方案;最后总结了现有的工作和描述了将来的工作方向。

## 2 相关工作

由于价格的低廉和广泛的应用,802.11 设备已经成了开发和评估新的无线系统和应用的事实上的选择[2],很多的实验平台,如 ORBIT[3], MOMENT Lab[4], CarTel[5], UCR-Testbed[6], Net-X[7] 都是建立在 802.11 硬件上的实验平台。通过软件仿真,硬件平台的测量,人们发现 IEEE 802.11 的 CSMA 机制在长距离网络、多跳网络、VoIP 支持等方面存在着缺陷,改进 IEEE 802.11 MAC 层,同时利用 TDMA 机制成了解决这些问题的一个重要方案。文献[8,9]为了解决长距离 WIFI 网络中 802.11 MAC 协议存在长距离链路采用 CSMA 易导致碰撞的问题,通过采用简单的 TDMA 机制来应对。文献[10-12]等在现有的 WiFi 网络中采用 TDMA 的接入方法来提高网络对并发视频,VoIP 业务等多媒体业务的支持数量。

使用 802.11 设备进行网络协议的开发实验有广泛的应用,同时在 802.11 设备上实现 TDMA 也在很多应用上有需求,但在面向工厂自动化应用对 TDMA 的要求(如时隙长度,同步精度)更为严格,其应用模式也有自己的特点(如短帧报文,周期性数据,突发性数据),现有的 TDMA 实现都根据自身面向的应用做出了相应的优化,不能直接应用到工厂自动化无线通信网络中来。

在面向工业应用的无线技术中,采用 CSMA 机制因为其随机冲突化解算法导致无法预测传输延迟,有可能造成很大的网络延时,甚至有可能在网络拥塞时暂时性地无法进行时间关键的数据交换,这在有很强的实时性要求的工业网络中是不能容忍的。TDMA 方案由于其传输的确定性,在大量节点场景下其冲突化解算法的优势,在工业无线应用中受到了越来越多的重视。在文献[13]中,提出了在 802.11e EDCA 机制上建立 TDMA 通信的方法,通过 TDMA 的方式规划实时节点的传输来规避实时节点之间的竞争,从而提高实时节点上实时数据实时性的同时能够与非实时节点兼容。ABB 公司的 WISA 系统[14]是目前适合面向工厂自动化应用的比较成熟的无线技术,也是目前唯一在工厂自动化中(车间级)商用的无线技术[15],它采用的正是 TDMA 的接入方式。

现在特别是基于 IEEE 802.11 物理层提出的针对工业应用的 MAC 层解决方案大多都是通过仿真方式验证的,缺少实验平台进行实验验证。

在实现方面尽管在 IEEE 802.11 协议中定义了基于 TDMA 的 PCF 规范,但在现有的产品几乎都没有相应的实现。因此很多研究机构对在 IEEE 802.11 硬件平台上实现 TDMA 方面开展了很多工作。TDMA 的实现最为重要的一个方面是对报文发送时间的精确掌控能力,在文献[16]中通过限制硬件缓冲区的报文个数来尽可能的掌控报文发送时间。文献[17]通过设置 QoS 及退避窗口参数的调整来达到相同的目的,它们本质上还是基于 CSMA 机制的,故需要将时间松弛地划分为时隙,需要预留较大的保护时间,因此有着较大的开销。而文献[2,8,9,18,19]等都是基于 Atheros 提供的在 Linux 操作系统下的驱动程序 MadWiFi[20],通过驱动程序提供的 API 控制射频芯片的报文发送时间,这种方式能够充分有效地利用硬件,能够屏蔽其缺省的 CSMA 机制。在 FreeBSD 8.0 中也开始在 802.11 协议栈中加入了一个仅支持 2 个站点的简单 TDMA 协议[21]。

现有的 TDMA 实现往往要求在一个时隙内可传输多个报文,要为最小的传输速率传输提供支持,同时一般都要求在时隙内实现 ACK 回复,所以它们都有着较长的时隙,如 WildNet 时隙长度为 10~40ms[8], MadMAC 时隙长度大于 20ms[2], FreeMAC 时隙长度为 25ms[18],这些都远远无法满足延时小于或等于 10ms 的要求。现在商业硬件上实现 TDMA 的工作大多基于 MadWiFi, MadWiFi 是 Atheros 芯片面向 Linux 驱动程序的一个过渡产品,现在已被 ath5k 取代,其硬件抽象层也只以二进制的方式提供,使得对硬件操控的能力受到限制。因此我们采用了对硬件更有控制力的 ath5k,充分挖掘软硬件的能力使传输时机的控制更为精确,同时针对工业控制报文的特点对时隙长度等 TDMA 参数进行优化使之满足工业应用需求。

## 3 TDMA-MAC 设计

### 3.1 针对工业应用的 TDMA-MAC 协议开发架构

我们的实现平台是建立在 Atheros 芯片 WiFi 网卡及其

在 Linux 下的驱动程序 ath5k 之上的,现有的基于 ath5k 的 802.11 协议栈架构如图 1(a)所示,其在链路层部分,一部分时间关键的 802.11 功能在 Atheros 芯片上实现,另一部分延迟容忍的 802.11 功能在 mac80211 中实现,而网络层传输层等高层协议都建立在 mac802.11 之上,以上部分的软件实现都是 Linux 内核态实现的,在用户态的应用程序通过 socket 等方式使用整个 Linux 802.11 子系统。为了使实验平台更适用于工业协议的开发,我们采用图 1(b)所示的开发架构。

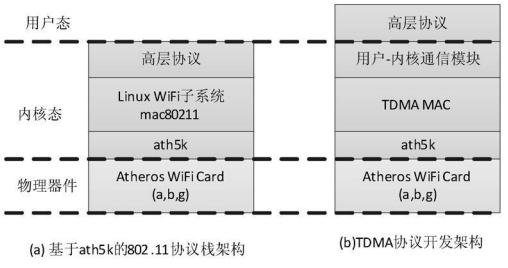


图 1　TDMA MAC 协议开发架构

与基于 ath5k 的 802.11 协议栈架构不同,我们的开发架构中在 MAC 层协议之上的高层协议通过用户-内核通信模块与 MAC 协议交互并工作于用户态。因为应用于车间级的通信系统往往拓扑结构相对简单,通信模式以集中式为主,所以其对网络层、传输层等的要求并不高,现场总线模型仅包括物理层、数据链路层、应用层和用户层,如 PROFIBU-DP 只包括物理层、链路层和应用层规范。通过这种方式的设计能够更快速灵活地开发和调试高层协议,宜于集中精力进行与实时性关系密切的 MAC 层协议的开发。

同时为了达到工厂自动化对实时性的要求,在 MAC 层去除了 Mac802.11 协议并采用了 TDMA 机制。在 MAC 协议的开发中通过禁止了 WiFi 网卡中 802.11 控制单元等方式,在 ath5k 之上实现了新的 MAC 层协议。

### 3.2　用户-内核通信模块

驱动程序 ath5k 及 TDMA MAC 协议的具体实现都是工作在核心态的,而很多的应用程序如工业现场数据仿真生成程序、现场数据收集与处理程序、诊断工具、配置工具等工作为了便于调试开发维护往往工作于用户态。因此需要开发用户-内核通信模块用于 MAC 协议与应用程序之间的交互,整个用户内核通信模块的功能结构如图 2 所示。

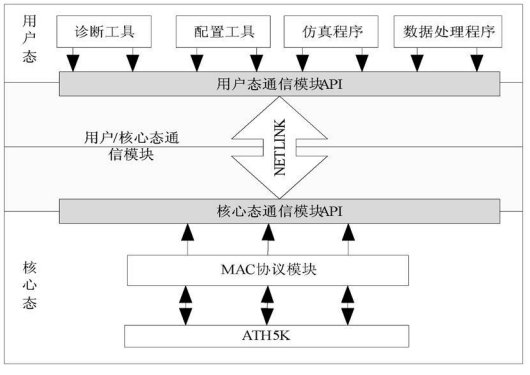


图 2　用户内核通信模块

整个模块的核心部分是基于 netlink 技术的,Netlink 是用于内核和用户空间进程,用户空间进程之间以及多个用户空间进程的组合与内核空间的进程间通信的类套接口机制。它适用于经常需要大数据量的进行内核与用户进程间传递数据的场合,同时它也可以用于硬件中断例程中。Netlink 具有可移植性、高度可扩展性同时支持基于事件通知的特点[22]及其对中断过程的支持,使得在 Linux 2.4 版及以后版本的内核中,几乎全部的中断过程与用户态进程的通信都是使用 netlink 套接字实现的,它将成为 Linux 用户态与内核态交流的主要方法之一。现在 netlink 技术在网络应用如 ip queue 工具、iproute[23]、无线 Mesh 路由协议 OLSDr[24]中都得到了广泛应用。

为此在核心态中,我们建立一个专门用于接收的内核线程负责所有来自用户态应用程序的处理,同时提供向用户态程序发送消息的 API,模块根据不同的消息类型以不同的广播组的形式广播给用户空间。在用户态,整个模块提供了一系列供应用程序使用的 API,应用程序利用 API 监听 netlink 广播组,不同的应用程序都能收到内核发来的消息,并且应用程序能够定制其感兴趣的消息类型,为用户应用程序的开发维护等提供了很好的灵活性。

## 4　TDMA-MAC 实现

TDMA 功能的实现主要包括两个方面:首先屏蔽 IEEE802.11 硬件的 CSMA 功能,实现对数据发送时机的精确控制等 TDMA 基本功能以完成 TDMA 状态切换的逻辑。其次,要构建一个 TDMA 网络需要利用硬件特性实现节点间高精度的时间同步。

### 4.1　TDMA 基本功能的实现

现有的在 IEEE 802.11 物理层上实现 TDMA 接入机制主要通过以下两类方式实现:首先是基于 802.11 MAC 实现:这种方式工作于现有的 802.11MAC 之内,在不要求修改现有驱动程序的情况下增强 MAC 层的功能,它通过修改 Linux 中 802.11 协议栈的相关参数来控制底层硬件发送数据的时机,并调度整个网络使数据发送不会发生碰撞,如文献[16]通过限制硬件缓冲区的报文个数来尽可能地掌控报文发送时间。文献[17]通过设置 QoS 及退避窗口参数的调整来达到相同的目的,但它们本质上还是基于 CSMA 机制的,它能够应用于现存的 802.11 硬件,能够很好地与链路层、网络层、传输层结合。但同时也有缺陷:受限于 802.11MAC 层提供的接口,相比硬件实现相同的功能要更大的开销,同时对硬件的控制比较低效,需要在软件上提供很多的保护机制增加了系统的开销。另一种方法是基于 MadWiFi 实现:这种方式大部分是工作在 Linux 与 Atheros 芯片上,通过利用 MadWiFi 驱动程序,以获得对传输时机精确时间调度及帧格式的灵活控制。这种方式能够充分有效地利用硬件,能够完全屏蔽其缺省的 CSMA 机制,现有的实现方式[2,8,9,18,19]基本上都是通过以下部分方式来共同完成这一工作的。

1. 禁止链路层自动 ACK
2. 禁止虚拟载波侦听
3. 禁止/控制随机退避过程
4. 帧结构控制
5. 禁止/使能每个报文传输成功中断
6. 每个报文的重传,传输功率及传输速率控制
7. 时间关键功能的可预测调度
8. 实时 Linux 系统

为了实现 TDMA,我们借鉴了现有在 MADWiFi 上的工

作,并最终采用了以下策略。

a)限制硬件缓冲区报文个数:因为当数据报文投递给硬件缓冲区之后,硬件接管整个报文的控制流程,如果硬件缓冲区同时有多个报文存在,软件将无法预测和确定或者控制每个报文的发送时机。硬件本身为支持 QoS(EDCA)服务有多个硬件缓冲队列(AR5212达 10 个),但在面向工厂自动化的工业应用中,数据报文短小,当节点本身有多个数据需要传输时,通过数据融合(将多个报文数据组合成一帧数据),不仅能够大大减小通信开销,同时节点本身数据不存在竞争服务的问题;不同节点间的数据我们可以通过 TDMA 的方式来规避竞争服务,因此在基于TDMA 的工厂自动化通信系统中,我们不充分利用硬件为 EDCA 机制而优化的硬件特性是可行的。在实际的实现中我们只用到了两个硬件缓冲队列,同时在使用时能够保证在同一时刻这两个硬件缓冲队列最多只有一个队列里有数据,为了达到这一要求有两种实现方式:

1. 在实现 TDMA 协议栈中限制发送队列的报文个数为1;

2. 在 TDMA 协议栈的定时器中断服务程序内只允许向硬件投递一个报文。

在具体实现上我们采用了后者,因为发送队列缓冲多个报文能够为以后数据融合等协议优化提供更高的灵活性。

b) 禁止硬件 CSMA 相关功能:在这方面主要分成以下几个部分。

1. 禁止虚拟载波侦听:虚拟载波侦听利用帧格式中的持续时间字段的保留信息实现虚拟侦听,当设备接收到 MAC 帧的持续时间字段,如果其值大于当前网络分配矢量(NAV)则更新值 NAV,NAV 就像一个计数器只有当其值到零时才会启动物理载波侦听。虚拟载波侦听会给发送传输增加明显的不确定因素,因此在 TDMA 实现时需要将其禁止。

2. 屏蔽物理载波侦听:在 IEEE 802.11 中,每个报文发送前都需要侦听信道一段时间,只有在检测到信道在侦听时间内一直空闲才发送数据,否则进入随机退避阶段。一旦进入随机退避阶段,我们将无法预测报文的发送时间,因此需要尽可能地禁止物理载波侦听。在 Atheros 硬件上还没有办法直接通过指令禁止射频芯片进行物理载波侦听,我们通过将 CCA(载波侦听)阈值设为一个极值来间接达到这种效果。在原来的工作过程射频芯片进行载波侦听,并将其对信道的采样值与 CCA 阈值进行比较,如果低于其值,则表示信道空闲,否则表明信道正忙。通过将 CCA 设为一个极大值能够使物理载波侦听失去效能,间接地屏蔽了载波侦听。

为了便于对 TDMA 状态逻辑加的控制,我们还进行了以下相关设置。

1. 退避过程控制,尽管经过禁止虚拟载波侦听,屏蔽物理载波侦听,禁止链路层 ACK 之后理论上不会再触发退避过程了,但还是将退避窗口的参数都设在了最小值。

2. 禁止 Beacon 中断,尽管在实验平台中很少甚至没有利用到 beacon,理论上不会触发 Beacon 中断,但还是将其禁止以免其的可能出现打断现有的 TDMA 调度。

3. 禁止链路层自动 ACK,采用链路层自动 ACK 后,射频需要为 ACK 进行等待重传等操作,一方面会将时隙的时间拉得比较长,另一方面如果 ACK 发生碰撞或者其它导致发送失败的情况将导致射频需要重新发送报文或者进入退避阶段,因此在实现时禁止链路层自动 ACK,我们在 TDMA MAC 模块中提供符合工业应用特点的 ACK 回复机制也在一定程度上涵盖了链路层自动 ACK 的功能,并且针对 TDMA 的工业应用需要优化了设计。

4. 禁止自适应速率变化,通过采用速率自适应算法能够通过平衡无线网络中物理层数据率与鲁棒性来优化网络性能,是一个很重要链路层机制。但在TDMA 机制中引入自适应速率算法会导致时隙长度需求的不确定性,给系统的调度带来了比较大的问题;并且在车间级的工业应用中传感器分布范围空间小密度大,大部分设备的移动都在接入节点的较强覆盖范围之内,因此我们在实验平台上禁止自适应速率变化,所有数据都采用统一的速率进行传输。

4.2 时间同步的实现

时间同步技术是TDMA 协议的一个重要基础,时间同步的精度直接反映在时隙中的保护边带上,时间同步精度低将增加系统开销降低系统的吞吐量,在有硬实时性要求的工业无线网络中,时间同步精度越低将使网络支持的节点个数越少。同时工业应用对时间同步在快速性、可靠性上提出了更高的要求:工业应用对设备上线时间,节点失效检测和恢复时间的高要求需要时间同步协议能够快速同步,并且迅速地从失同步状态恢复到同步状态,同时工业应用对通信系统高可靠性的要求也需要时间同步能够对故障快速检测并恢复。

IEEE 802.11 的时间同步(TSF)机制分为两类:在基础设施的 BSS 网络中,AP 作为整个网络的同步源,周期性广播包含 TSF 定时器时戳的信标帧,同一 BSS 网络中的站点接收到信标帧后,将信标帧里的时戳与本地维护的TSF 定时器值比较,如果两者不同则将本地 TSF 定时器设为接收到的时戳值;在 IBSS 网络中每个站点维护一个 64 位 TSF 定时器,时间同步通过节点交换同步信标帧来实现的,所有的站点都在一个 TBTT(目标信标帧传输时间)来发送,在每个 TBTT 开始后每个站点随机一定的延迟以发送同步帧。在这个延迟内如果收到同步帧,将取消其同步帧发送,且如果收到的同步帧时戳大于当前自己的同步定时器的值,将用同步帧里的时戳更新同步时器的值,否则将其丢弃;如果在延迟时间内没有收到同步帧,延迟结束时将自己同步定时器的值写入同步帧内并将其发送。

因为实验平台暂时只支持星形的拓扑结构,因此同步算法采用了和 IEEE 802.11 中 BSS 网络同步相类似的方法。为了支持 IEEE 802.11 的同步算法,Atheros 硬件能够在发送时给信标帧自动打上硬件时戳,在接收时自动为每一个报文记录接收时 TSF 定时器的低 32 位。因为工业应用中报文的长度非常小,往往仅有几个字节的数据,为了减少协议的通信开销,需要重新定义帧格式,同时要实现较高的同步精度必须充分利用硬件的相关特性,尽量减少软件计算,插入时戳等操作带来的不确定延时,但是 802.11 硬件在实现时针对802.11协议做了相应的优化,因此在实现时需要着重解决以下两个问题。

1. 硬件在信标帧发送时会自动给报文打上时戳,由于是针对 802.11 协议其时戳存放的位置为报文的 24~ 31 个字节,但工业应用中报文往往很小,而采用较小的报文将无法利用硬件自动插入时戳的功能,易影响同步精度。

2. 重新定义帧格式之后,硬件不能识别重新定义后的信

标帧,导致站点无法在收到信标帧后根据信标帧里的时戳自动维护本地的 TSF 定时器。

为了解决较小的报文将无法利用硬件自动打时戳的功能的问题,在 AP 端采用数据融合方式,其基本过程如图 3 所示。当 AP 到达其发送时隙时,它将所有需要发送的报文组合成一帧(如果过大也有可能多帧),并向全网广播此帧数据,节点接收数据后,截取并解析发给自己的数据段。如果需要响应 ACK,则在节点的发送时隙向 AP 发送 ACK。

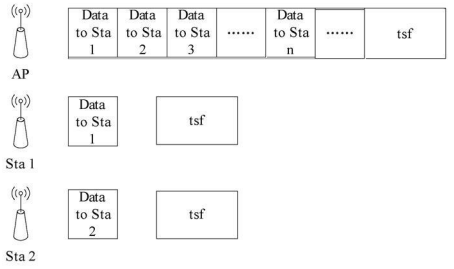


图 3　下行数据融合

由于在自动化控制中报文长度较小,物理层前导码和 PLCP 帧头的传输所占的开销较大,比如在 54Mbps 的速率中,传输一个负载长度为 40byte 的报文,物理层前导码和 PLCP 帧头的传输时间为 20us,而整个数据传输所需要时间为 25.9us,物理层前导码和 PLCP 帧头的开销约为 77.2%。在 AP 端通过数据融合能够大大降低物理层前导码和 PLCP 帧头的开销,在工业无线应用[14],VoIP 应用[12]中都采用了类似的技术来减少物理层通信开销的比重。采用这种方法后下行(AP 发送至节点)时隙长度需要比上行时隙长度更大,其发送报文的长度也可以相应较长,当用于发送信标帧时,信标帧可以融合其它数据,整个信标帧的长度可以达到 32 个字节,因此可以利用上硬件自动给信标帧报文添加时戳的功能,解决软件打时戳影响同步精度的问题。

重新定义帧格式后,硬件无法识别新定义的信标帧,因此无法利用硬件自动维护 TSF 定时器的功能,在此我们采用软件的方法来维护本地 TSF 定时器。在接收到数据报文后,我们能够获取报文的 64 位接收时戳信息,其高 32 位是 ath5k 驱动程序添加的,低 32 位是由硬件自动在接收到报文时对本地 TSF 定时器低 32 位的一个快照,其精度是非常高的。因此在接收到信标帧之后,将信标帧里的时戳与接收时戳进行比较,并将两者的差值补偿给本地 TSF 定时器,通过这种方式实现整个时间同步的过程。

## 5　TDMA-MAC 的实验验证

### 5.1　TDMA 基本功能实验验证

为了验证 TDMA 基本功能实现方法的有效性,我们采用以下的实验方法:在两台 PC 机上均采用 TP-LINK WN550G 无线网卡(射频芯片采用的是 Atheros 的 Ar5212),操作系统为 Debian Lenny,内核为 Linux 2.6.26 同时配备以 ath5k 为基础的实验平台的相关软件及应用程序,其中一台 PC 工作在 CSMA 模式以 2ms 为周期发送总长度为 40 个字节的数据报文,此 PC 作为干扰源专用于与测试 PC 竞争信道,干扰测试 PC 的数据传输。测试 PC 也以 2ms 为周期发送总长度为 40 个字节的数据报文,以不同的模式发送数据各约 11000 个,以报文加入驱动缓存起到触发发送。完成中断之间耗费的时间定义为发送所需时间,在不同模式下发送所需的时间如图 4 所示。

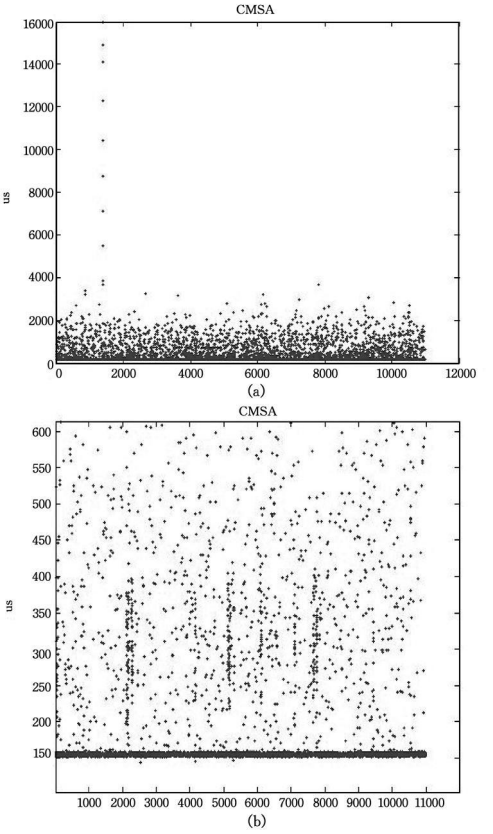


图 4　CSMA 模式数据发送时间消耗(a:CSMA 模式下数据发送时间;b:CSMA 模式下数据发送时间局部细节)

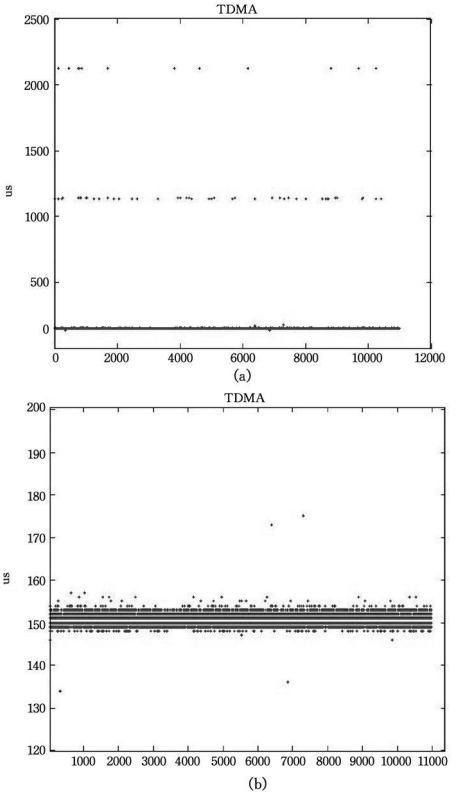


图 5　TDMA 模式数据发送时间消耗(a:TDMA 模式下数据发送时间;b:TDMA 模式下数据发送时间局部细节)

从图 4、图 5 中可以看出,当设备采用 CSMA 机制发送数

据时,由于碰撞退避等原因发送所需要时间有很大的随机性,同时大约 21.5% 的数据发送需要大于 200us。而当设备采用 TDMA 机制发送数据时,数据发送需要的时间具有一定的确定性,99.5% 的报文发送所需要的时间在 140us ~ 160us 之间,说明采用 TDMA 机制后发送时间并没有受到干扰源的影响,证明了我们通过禁止 CSMA 相关功能和限制硬件缓冲区个数的方法能够满足 TDMA 对发送时间确定性的要求。

从图 5 中可以发现现有的机制能够实现一个简单毫秒级时隙调度的 TDMA 系统,但如果要实现微妙级的时隙调度还存在以下两个问题:1. 在 TDMA 模式下数据发送时间中有大约 0.5% 的报文发送所需时间分布在约 1.15ms 与 2.15ms 上,尽管报文数量不多但在精确的工业硬实时系统中还是有可能带来可靠性问题。2. 理论上以 48Mbps 的速率发送 40 字节的数据只需要约 27us 的时间(发送物理层和 PLCP 头部需要 20us,发送数据需要约 7us),而实验结果表明从数据进入驱动缓冲区到发送完成需要大约 150us 的时间,这意味着大约 80% 的传输时间开销用于节点处理必不可少的软件处理开销上,使得时隙为补偿软件延时而导致的开销很大,这将严重影响系统的吞吐量。针对这类软件延时的问题,我们将来拟采用文献[25] 的流水线传输方式,来提高信道利用率和系统的吞吐量。

发送所需时间出现少量的较大值,可能有两个原因:1) 中断导致延时,因为 Linux 系统本身不是一个实时操作系统,同时测试 PC 运行的图形界面系统可能在某些时刻系统负荷较大,导致发送程序被暂时中断。2) 当射频芯片正在接收信号时发送数据,射频状态机不会从接收立即切换到发送,这在一定程度上也会给发送时间带来一定的不确定性,但在一个纯的 TDMA 系统中由于系统都工作在无竞争状态,所以这种情况不会发生。在将来的工作中需要进一步确定导致发送所需时间出现少量较大值的原因,使系统能够准确预测数据发送完成时间。

### 5.2 时间同步实验验证

为了验证时间同步方法的有效性,我们进行了以下实验:一个主设备周期性地发送信标帧,信标帧中含有主设备发送信标帧时的 TSF 的值。从设备接收信标帧,并记录接收时本地 TSF 的值,理想情况下(即同步无误差),本地 TSF 的值应与信标帧中的 TSF 时戳值相等,我们将两者间的差值定义为同步误差,在实验过程中每收到 16 个信标帧才进行一次同步操作,其余的信标报文用于跟踪同步误差。整个实验,对每个不同的同步周期(TSP, time synchronization period) 均采样 10000 个同步误差值,这个过程中进行了大约 625 次同步,同步周期分别为 160ms 与 3.2s 的实验结果如图 6 所示。

从实验结果来看,当同步周期为 160ms 时,99% 的时间内同步误差少于或等于 12us;当同步周期为 3.2s 时,99% 的时间内同步误差小于或等于 14us,即使同步周期为 3.2s,同步算法的同步精度也达到了 20us。以上数据说明该同步算法精度是比较高的,能够让毫秒级时隙的 TDMA 系统中由于弥补同步误差的保护边带所占用的开销保持在一个很小的水平上。

尽管同步算法取得较高的精度,但是其还有以下两点需要在后续的工作中进行改进:1) 为了使 TSF 同步尽可能精确,硬件对信标帧接收和发送的延时都做了补偿,但其延时补偿都是针对 6Mbps 的速率下进行的,当采用不同的速率(如实验采用的是 48Mbps) 时需要通过其它手段进行相应的补偿。2) 硬件中 TSF 的同步精度为 ±2us,其数值以 2 步进,故在实验结果中同步误差的差值都为偶数,要进一步提高同步精度需要提高硬件 TSF 定时器的精度。

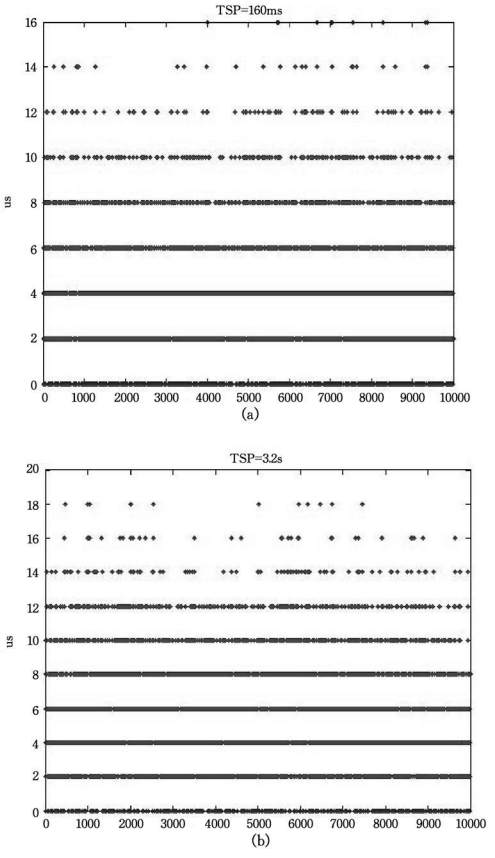


图 6 时间同步算法结果(a: 同步周期为 160ms 时的同步结果; b: 同步周期为 3.2s 时的同步结果)

结束语 本文建立了一个在商用 802.11 硬件上面向工厂自动化的基于 TDMA 的无线网络研究平台。通过在 ath5k 之上进行编程,能够在可配置的时间里发送报文,并且能够灵活控制帧格式,实现高精度的时间同步,用户能够据此编写适合工业应用的无线通信协议。在本文中,我们还建立了一个基于 TDMA 的面向工业应用的高速无线网络原型系统,实验结果表明其完全能够支撑毫秒级时隙调度的 TDMA 应用。

将来的工作首先是实现微秒级的时隙调度,它要求系统工作在实时 Linux 上;并需要对流水线传输的方案予以实现以提高系统吞吐量;同时还需要通过补偿硬件延时和提高 TSF 定时器精度等方法进一步提高时间同步算法的精度,最终使实验平台成为能够满足较为宽松的面向工厂自动化应用的通信平台的需求。

## 参 考 文 献

[1] Pellegrini D F, Miorandi D, Vitturi S, et al. On the use of wireless networks at low level of factory automation systems[J]. Industrial Informatics, IEEE Transactions on, 2006, 2(2): 129-143

[2] Sharma A, Tiwari M, Zheng H. MadMAC: Building a Reconfiguration Radio Testbed using Commodity 802.11 Hardware[C]// Networking Technologies for Software Defined Radio Networks SDR. 2006: 78-83

(下转第 344 页)

## 参 考 文 献


[1] 韦乐平. 城域电信级以太网的特征与新发展[J]. 电信科学, 2007(2)

[2] 刘韵洁, 张云勇, 张智江. 下一代网络服务质量技术[M]. 北京: 电子工业出版社, 2005

[3] 曾华燊. 现代网络通信技术[M]. 成都: 西南交通大学出版社, 2003

[4] Allan, Bragg, McGuire, et al. Ethernet as Carrier Transport Infrastructure[J]. IEEE Communications Magazine, February 2006

[5] 张奇智, 尹汝波. 交换式工业以太网的现状和研究[J]. 传感器世界, 2005, 2: 34-39


(上接第 304 页)


[3] orbit[EB/OL]. http://www.orbit-lab.org/

[4] Moment[EB/OL]. http://moment.cs.ucsb.edu/index.html

[5] Hull B, Bychkovsky V, Zhang Y, et al. CarTel: a distributed mobile sensor computing system[C]// Proceedings of the 4th international conference on Embedded networked sensor systems. Boulder, Colorado, USA, 2006: 125-138

[6] Broustis I, Eriksson J, Krishnamurthy S V, et al. A Blueprint for a Manageable and Affordable Wireless Testbed: Design, Pitfalls and Lessons Learned[C]// Testbeds and Research Infrastructure for the Development of Networks and Communities. TridentCom, 2007: 1-6

[7] Chereddi C, Kyasanur P, Vaidya N H. Design and implementation of a multi-channel multi-interface network[C]// Proceedings of the 2nd international workshop on Multi-hop ad hoc networks. Florence, Italy, 2006: 23-30

[8] Patra R, Nedevschi S, Surana S, et al. Wildnet: Design and implementation of high performance wifi based long distance networks[C]// 4th USENIX Symposium on Networked Systems Design and Implementation. Cambridge, MA, USA, 2007: 87-100

[9] Dhekne A, Uchat N, Raman B. Implementation and Evaluation of a TDMA MAC for WiFi-based Rural Mesh Networks[C]// 3rd ACM Workshopo on Networked Systems for Developing Regions(NSDR). Montana USA, 2009

[10] Kandhalu A, Rowe A, Rajkumar R R, et al. Real-Time Video Surveillance over IEEE 802.11 Mesh Networks[C]// IEEE Real-Time and Embedded Technology and Applications Symposium(RTAS). San Francisco USA, 2009: 205-214

[11] Guo F, Chiueh T. Software TDMA for VoIP applications over IEEE802.11 wireless LAN[C]// IEEE International Conference on Computer Communications (INFOCOM). Alaska USA, 2007: 2366-2370

[12] Verkaik P, Agarwal Y, Gupta R, et al. SoftSpeak: Making VoIP Play Well in Existing 802.11 Deployments[C]// USENIX Symposum on Networked Systems Design and Implementation (NSDI). Boston, USA, 2009

[13] Costa R, Portugal P, Vasques F, et al. A TDMA-based mechanism for real-time communication in IEEE 802.11e networks[C]// Emerging Technologies and Factory Automation (ETFA). Bilbao, Spains, 2010: 1-9

[14] Scheible G, Dacfey D, Endresen J, et al. Unplugged but connected-Design and Implementation of a Truly Wireless Real-Time Sensor/Actuator Interface[J]. Industrial Electronics Magazine, IEEE, 2007, 1(2): 25-34

[15] Korber H J, Wattar H, Scholl G. Modular Wireless Real-Time Sensor/Actuator Network for Factory Automation Applications[J]. IEEE Transactions on Industrial Informatics, 2007, 3(2): 111-119

[16] Rao A, Stoica I. An overlay MAC layer for 802.11 networks[C]// The International Conference onMobile Systems, Applications, and Services (MobiSys). Washington, USA, 2005

[17] Djukic P, Mohapatra P. Soft-TDMAC: A Software TDMA-Based MAC over Commodity 802.11 Hardware[C]// IEEE International Conference on Computer Communications (INFOCOM). Rio de Janeiro, Brazil, 2009: 1836-1844

[18] Sharma A, Belding E M. FreeMAC: framework for multi-channel mac development on 802.11 hardware[C]// Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow. Seattle, WA, USA, 2008: 69-74

[19] Neufeld M, Fifield J, Doerr C, et al. SoftMAC: A Flexible Wireless Research Platform[C]// Workshop on Hot Topoics in Networks(HotNets). Maryland, USA, 2005

[20] MadWiFi[OL]. http://madwifi-project.org/

[21] Leffler S. TDMA for Long Distance Wireless Networks[OL]. people.freebsd.org/~sam

[22] Neira-Ayuso P, Gasca R M, Lefevre L. Communicating between the kernel and user-space in Linux using Netlink sockets[J]. Software: Practice and Experience, 2010, 40(9): 797-810

[23] iproute2[EB/OL]. http://www.linuxfoundation.org/collaborate/workgroups/networking/iproute2

[24] OLSRd. OLSRD: Ad-hoc Wireless Mesh Routing Daemon[EB/OL]. http://www.olsr.org

[25] Chintalapudi K K, Venkatraman L. On the Design of MAC Protocols for Low-Latency Hard Real-Time Discrete Control Applications over 802.15.4 Hardware[C]// Proceedings of the 7th international conference on Information processing in sensor networks. 2008: 356-367