****

硕士学位论文

|  |
| --- |
| 车载环境下媒体接入控制协议优化设计 |

|  |  |
| --- | --- |
| 作者姓名 | 陈钦波 |
| 学科专业 | 电路与系统 |
| 指导教师 | 吴宗泽副教授 |
|  |  |
| 所在学院 | 电子与信息学院 |
| 论文提交日期 | 2014年5月 |

**Media Access Control Protocol Optimization in Vehicle Environment**

A Dissertation Submitted for the Degree of Master

**Candidate：Chen Qinbo**

**Supervisor：Wu Zongze**

South China University of Technology

Guangzhou, China

**分类号：TP393.09 学校代号：10561**

**学号：201120107536**

华南理工大学硕士学位论文

**车载环境下媒体接入控制协议优化设计**

作者姓名：陈钦波 指导教师姓名、职称：吴宗泽副教授

申请学位级别：工学硕士 学科专业名称：电路与系统

研究方向：嵌入式与SOC

论文提交日期：2014年6月5日 论文答辩日期：2014年6月1日

学位授予单位：华南理工大学 学位授予日期： 年 月 日

答辩委员会成员：

主席： 吴宗泽副教授

委员： 谢胜利教授 傅予力教授 周智恒副教授 李波副教授

**华南理工大学**

**学位论文原创性声明**

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律后果由本人承担。

作者签名： 日期： 年 月 日

**学位论文版权使用授权书**

本学位论文作者完全了解学校有关保留、使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属华南理工大学。学校有权保存并向国家有关部门或机构送交论文的复印件和电子版，允许学位论文被查阅（除在保密期内的保密论文外）；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。本人电子文档的内容和纸质论文的内容相一致。

本学位论文属于：

□保密，在年解密后适用本授权书。

□不保密,同意在校园网上发布，供校内师生和与学校有共享协议的单位浏览；同意将本人学位论文提交中国学术期刊(光盘版)电子杂志社全文出版和编入CNKI《中国知识资源总库》，传播学位论文的全部或部分内容。

(请在以上相应方框内打“√”)

作者签名： 日期：

指导教师签名： 日期：

作者联系电话： 电子邮箱：

联系地址(含邮编)

摘 要

随着互联网技术和交通系统的不断发展，车联网（IOV，Internet of Vehicles）受到越来越多研究机构和厂商的关注和研究。车联网是当今研究前沿物联网（IOT，Internet of Things）的一个集合，是物联网应用于智能交通系统（ITS，Intelligent Transport System）领域的具体形式。车联网是指汽车高速移动的环境下，车载终端节点通过车辆上的无线收发装置，以及路边固定终端节点的无线收发装置，按照特定通信协议组合构建连接起来形成的一个车用互联网。车联网涉及到数据处理技术、无线电技术、通信技术、传感技术、自动控制技术以及信息发布技术等各种技术，涉及面广。车联网旨在提供一个车-车、车-路高速的数据通信方式，实时准确智能地整合车载相关数据信息，解决城市交通拥堵问题，确保行车安全，同时提供其他的商务娱乐应用服务，最终实现智能交通、智能城市管理和智能城市信息服务。

汽车做为车载环境下的移动媒体，需要遵循特定的接入控制规则以接入车联网。现有成熟的媒体接入控制（MAC，Media Access Control）主要是国际电子电气工程师协会IEEE802.11无线技术，采用载波侦听多点接入/避免冲撞（CSMA/CA，Carrier Sense Multiple Access with Collision Avoidance）的媒体访问控制机制。CSMA/CA使用了物理载波监听和虚拟载波监听来尽量避免冲撞，当终端节点发送数据检测到无线载波链路冲突时，必须退避一个随机时间再尝试发送。除了CSMA/CA控制机制外，还有时分多址访问技术（TDMA，Time Division Multiple Access）机制，按照一定的映射算法给各个终端节点分配不同时间片的方式调度整个系统中各个终端节点的数据通信。

本论文在调查研究无线网络媒体接入控制技术和车联网相关技术的基础上，研究WAVE协议和IEEE1609协议族，并在Linux操作系统编程和Linux内核网络子系统的基础上，借助Linux开放的网络协议栈接口，实现一个初步的双网卡媒体控制接入系统平台。同时还研究了TDMA中的时间片映射分配，提出位置辅助随机访问方案，利用遗传算法通过仿真验证了该方案相比普通的随机访问方案有着较好的性能。

**关键词**： 车联网；媒体接入控制；WAVE；802.11p ；CSMA/TDMA；Linux网络编

Abstract

With the progressive development of the Internet technology and the transport system , research institutes and manufacturers have aroused more and more concern and study about the Internet of Vehicles(IOV). The Internet of Vehicles is a subset of today's the leading edge of scientific research, also is the concrete format used in Intelligent Transport System(ITS) of the Internet of Things(IOT).

The Internet of Vehicles is a vehicle–use network at high speed vehicle environment, which is formed by the high speed moving vehicle terminal node with the wireless transceiver, and a stationary roadside terminal node with the wireless transceiver, constructed by communications protocol. The Internet of Vehicles comes to a wildly and variety of technologies such as data processing technology, wireless networking technology, communications technology, sensor technology, automatic control technology, information dissemination technology and so on. The Internet of Vehicles is intended to provide a high speed data communication between vehicle-vehicle, vehicle-roadside, integrate the vehicle information real-timely, accurately and intelligently, solve the urban traffic congestion problems, ensure road safety, also provide additional business and entertainment applications services, and ultimately achieve the goal of intelligent transport, management the city smartly and intellective city information services.

The automotive vehicle must follow a specific vehicle media access control rule as a movable media to access the network . Currently the mature media access control is mainly the Institute of Electrical and Electronics Engineers (IEEE)802.11 wireless Media Access Control technologies using CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) . CSMA / CA uses the physical carrier sense and the virtual carrier sense to try to avoid the collision, when a terminal node transmits data carrier links conflict happen, it should retreat a random time before it tries to send. In addition to the CSMA/CA control mechanism, TDMA (Time Division Multiple Access) is another choice, which according to a mapping algorithm to assign different time slots for each terminal node, to schedule the data communication of every terminal node in the entire system.

This thesis research about the WAVE wireless protocol and IEEE1609 protocol, and with the Linux operating system programming, the Linux kernel networking subsystem, and the Linux open network protocol stack interface, achieve a simple dual NIC media access control system platform. It also studies the TDMA time slot allocation map of Location-Assisted Random Access Scheme, which is verified to be a better performance by simulation compared to normal random access scheme.

**Keywords:** Internet of Vehicles; WAVE; 802.11p; TDMA; Linux network program

目录

[摘 要 I](#_Toc387826485)

[Abstract II](#_Toc387826486)

[目录 IV](#_Toc387826487)

[第一章 绪论 1](#_Toc387826488)

[1.1 研究背景及意义 1](#_Toc387826489)

[1.2 国内外研究现状 3](#_Toc387826490)

[1.3 选题的目的及意义 6](#_Toc387826491)

[1.4 论文主要研究内容及结构安排 7](#_Toc387826492)

[第二章 相关技术 8](#_Toc387826493)

[2.1 车联网相关技术概述 8](#_Toc387826494)

[2.2 IEEE802.11p 9](#_Toc387826495)

[2.2.1 IEEE802.11协议 10](#_Toc387826496)

[2.2.2 IEEE802.11p 16](#_Toc387826497)

[2.3 IEEE1609协议栈 17](#_Toc387826498)

[2.4 Linux相关编程 21](#_Toc387826499)

[2.4.1 Linux的特点 22](#_Toc387826500)

[2.4.2 Linux体系结构 25](#_Toc387826501)

[2.4.3 Linux设备驱动程序 26](#_Toc387826502)

[2.5 本章小结 27](#_Toc387826503)

[第三章 基于双网卡的媒体接入控制优化 29](#_Toc387826504)

[3.1 方案描述 29](#_Toc387826505)

[3.2 相关软件代码实现 32](#_Toc387826506)

[3.2.1 libpcap移植 32](#_Toc387826507)

[3.2.2 libpcap抓包收发示例 34](#_Toc387826508)

[3.2.3 双网卡平台系统 36](#_Toc387826509)

[3.2.4 性能测试 38](#_Toc387826510)

[3.2.5 改进 41](#_Toc387826511)

[3.3 IEEE802.11p驱动移植概述 41](#_Toc387826512)

[3.4 本章小结 43](#_Toc387826513)

[第四章 基于TDMA的媒体接入控制设计 44](#_Toc387826514)

[4.1 TDMA概述 44](#_Toc387826515)

[4.2 位置信息辅助的多路访问方案 46](#_Toc387826516)

[4.3 冲突概率和吞吐量 48](#_Toc387826517)

[4.3.1 冲突概率 48](#_Toc387826518)

[4.3.2 吞吐量 51](#_Toc387826519)

[4.4 建模求解 51](#_Toc387826520)

[4.5 仿真结果 54](#_Toc387826521)

[4.6 本章小结 57](#_Toc387826522)

[总结与展望 58](#_Toc387826523)

[参考文献 59](#_Toc387826524)

[攻读硕士学位期间取得的研究成果 61](#_Toc387826525)

[致谢 62](#_Toc387826526)

1. 绪论
   1. 研究背景及意义

当今世界，经济不断发展，科技不断提高，信息技术己经融入到生活的方方面面。把信息技术融入到生活以及工业技术当中，可以提升生活质量、提高生产效率。如果世界上的所有物体都纳入同一个信息网络中，那将形成一个比互联网更加庞大的网络，其对人类社会的发展将是革命性的进步，于是物联网编号项[1]（Internet of Net）编号项的概念诞生了。物联网指的是将一切和人类相关的生活科技生产教育交通等“智能化物件或动物”或“智能尘埃”，通过各种有线/无线的短距离/长距离通讯网络实现互通互联、应用融合、以及基于云计算的营运等模式，提供安全可控、个性化的报警联动、实时在线监测、远程控制、决策支持、定位追溯、调度指挥统计报表、远程维保、预案管理、安全防范、在线升级、领导桌面等管理和服务功能，实现对“万物”的“高效、节能、安全、环保”的“管、控、营”一体化。物联网是世界发展的一种趋势，这一技术将会发展成为一个上万亿规模的新兴信息产业。面对物联网潜在的巨大经济效益，美国、欧盟、日本等发达国家和地区相继制定了各自的物联网技术与发展计划，我们物联网技术研究市场开拓也紧随国际步伐。IBM首先提出了“智慧地球、物联网和云计算”[2]，前几年温家宝总理提出“感知中国”计划，明确物联网技术及产业为国家战略发展产业，物联网技术与产业在我国迎来了蓬勃发展的良好机遇[3]。交通运输是现代社会发展和国民经济的基础，现代社会城市化速度的加快、全球经济—体化进程、国民经济的高速增长的加快以及人们对交通的需求越来越高，交通系统所涉及的领域和范围越来越广，传统的交通管理方式显得力不从心。利用科技信息技术，引入实现准确、实时、全方位的交通信息采集、传输、处理、发布与控制，保障交通安全、高效的智能交通系统ITS[4]成为提高交通管理水平的重要途径。智能交通系统ITS是由车辆控制系统、交通监控系统、运营车辆管理系统、交通信息发布系统等集成一体，面向交通运输领域服务的现代电子信息系统，其将道路、驾驶员和车辆有机地结合在一起，利用对车辆、驾驶员和道路实时信息采集、自动智能处理和控制来达到充分利用交通资源的目的[5]。

车联网是当今研究前沿物联网（IOT，Internet of Things）的一个集合，是物联网应用于智能交通系统（ITS，Intelligent Transport System）领域的具体形式。车联网是指汽车高速移动的环境下，车载终端节点通过车辆上的无线收发装置，以及路边固定终端节点的无线收发装置，通过通信协议组合构建连接起来形成的一个车用互联网。车联网涉及到数据处理技术、无线技术、通信技术、传感技术、自动控制技术以及信息发布技术等各种技术，涉及面广。车联网旨在提供一个车-车、车-路高速的数据通信方式，实时准确智能地整合车载相关数据信息，解决城市交通拥堵问题，确保行车安全，同时提供其他的商务娱乐应用服务，最终实现智能交通、智能城市管理和智能城市信息服务[6]。世界上现阶段车联网大范围使用的公共移动通信网络是3G网络。3G网络有着较好的网络覆盖范围和数据通信速率，但其接入网络和通信方法的限制，仍无法满足网络拓扑快速变化的车载环境。因此协议栈WAVE[7]（Wireless Access in Vehicular Environment）应运而生，由IEEE制定，专门用于ITS中基于无线的通信应用[8]。WAVE和3G技术的主要参数对比如表一1所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 项次 | 参数名称 | WAVE | WCDMA | TD-SCDMA | CDMA2000 |
| 1 | 频段 | 5800MHz | 850、900、1800、1900MHz | | |
| 2 | 速率 | 3~27Mbps | ≤3Mbps | ≤1.28Mbps | ≤3.6864Mbps |
| 3 | 双工方式 | FDD、TDD | FDD、TDD | TDD | FDD、TDD |
| 4 | 复用技术 | OFDM | CDMA | CDMA | CDMA |
| 5 | 调制方式 | BPSK、QPSK、  16QAM、64QAM | QPSK、BPSK | QPSK、BPSK | QPSK、BPSK |
| 6 | 移动性 | ≤120km/h | 较低 | | |

表一1WAVE与3G主要参数比较表格

基于WAVE技术的应用，可涉及到交通拥堵治理、交通安全、公共服务相关领域和商业应用等相关领域。其中交通拥堵治理和交通安全是车联网中最基本最重要的应用，因此WAVE协议栈中开辟了专用通信信道用于这些方面的应用。

* 1. 国内外研究现状

车联网作为智能交通系统的重要组成部分，是各国智能交通发展研究中的关键部分，对提高交通运输系统的安全性和通行效率具有十分重要的作用。以美国、欧洲和日本为例，推出相关项目研究车路协同系统。在美国、欧洲和日本的ITS发展经历了三个主要阶段，1930-1980年为研究的准备阶段，1980-1995年为可行性研究的阶段，从1995年后以产品开发为主。各国都开展DSRC[9]标准的制定工作，美国将5.9GHz频段作为专用于车辆通信环境，在欧洲也制定车路协同的标准和规范。目前，国际上推动ITS发展的三大组织是美国的ITS America、日本的VERTIS和欧洲的ERITCO。[10]

美国是世界上最早进行ITS研究的国家。在上个世纪，美国的电子路线引导系统（Electronic Route Guidance System，ERGS）已经开展实施。在美国多个地方研究智能交通，AHS项目由国家自动公路系统会社推行。在近几年美国的研究包括车辆与车辆间的安全性研究、车辆与路侧基础设施间的安全性、实时的数据捕获、动态地移动应用、道路天气和环境研究、车路协同等。美国交通部最新发布的五年计划“Connected Vehicle”，致力于提高全美交通行车的安全性和机动性。该五年计划基于无线电通信技术，连通汽车、卡车、公共汽车和地铁系统，使这些交通设备及其它配套的设施能够共享交通安全和流动信息，并最终达到拯救生命、减少伤害、缓解拥堵和改善环境的目的。[11]

日本的ITS最早从上世纪80年代末开始，进行ARTS（Advanced Road Transportation Systems）、VICS（Vehicle Information and Communication System）和AVS（Advanced Safety Vehicle）项目研究，日本早期主要研究集中于安全领域，近期则注重环保安全绿色。1996年开始的VICS系统为核心研究系统，到目前已经基本完成智能交通系统的基础和核心技术的开发阶段，进入实用技术的研发，其中VICS项目主要促进研究和发展车辆安全技术，已形成了庞大的产业和成熟的产品。学术研究和工业组织的代表项目为VERTIS（Vehicle Road and Traffic Intelligent Society）。日本政府在2006年1月，将实现世界上最安全高效的道路交通计划作为其新一代IT改革战略的重要组成部分。在该计划中日本政府承诺在未来的数年内实现将的每年因交通事故死亡人数减少不到5000的目标[12]。2005-2010期间主要研究车路间协调、智能汽车系统等；在2010年后，日本将重点研究车路协同系统的实际应用，构建人车路三者一体化的信息网络和安全控制技术。

欧盟作为欧洲的一体化组织目前的车联网研究项目分为两大类，一个是政府主导的公共服务项目，另一类是民间组织对车载自组织网络的研究项目在1930至1980年期间，是欧洲智能交通发展的第一个阶段，这个阶段的技术发展并不成熟。在1980至1995这个阶段中，欧洲的19个国家的政府、公司和教育机构建立了欧洲高效安全交通系统计划（PROMETHEUS)，在1987至1994年间发展了DRIVE计划。在PROMETHEUS计划下开展了ARGO，设计、开发和研究交通信息化领域的创新的解决方案，实现交通的协调和管理。在1991年成立ERTICO以促进智能交通的不断发展，包括eSafety Forum和EasyWay来加强ITS的实施和应用。在2010年制定新的框架Directive 2010/40/EU，框架的制定加速创新的交通技术在欧洲的部署，解决ITS在欧洲的兼容性和互操作性。除了上术的两大研究性计划，欧洲各国也都在开展适合本国交通运输环境的车联网研究项目，将车路协同通信作为重点的研究方向。

对比国外发达国家，我国智能交通管理的发展起步较晚，因而相关产业的发展也比较落后。直到90年代后相关的研究部门才开始研究ITS发展战略和GIS（Geographic Information System，地理信息系统）、GPS（Global Positioning System，全球定位系统）、EDI（Electronic Data Interchange，电子数据交换）在交通中的应用等，重视交通信息网络的建设，如交通部的公路智能运输系统发展战略研究、铁道部TMIS（Transportation Management Information System，铁路运输管理系统）、DMIS（Dispatch Management Information System，调度指挥管理系统）等信息系统开发。虽然现在与发达国家相比，我们的车联网系统还有很大的差距，但是因为随着信息技术、电子技术、传感通信技术和计算机等技术的快速发展，我国从上世纪80年代开始加快对智能交通系统的技术研究，近几年物联网的发展对我国车联网的发展起推动作用。经过多年的努力在国内车联网的发展上我们也已取得明显的进展。但是我国智能交通系统中目前所使用的技术仅能满足一些有限的应用，如ETC（Electronic Toll Collection，电子不停车收费系统）等的要求。

我国在车联网络的研究方面还有很长的路要走，还有更多的工作需要做。我们不能仅仅满足于ETC、小区门禁这类简单的应用，对于大容量交通信息的实时传输、交通信息的实时发布、基于实时交通状况的车辆导航和交通安全等复杂应用等等这些新的高级需求，需要我们持续努力、完善以及变革创新。

因此我们车联网的发展面临巨大的挑战和机遇：

* + 我们经济发展较快，高速公路发展迅速，规摸不断扩大，纳构不断复杂，运营管理难度不断增加，使得对高效稳定的智能交通管理手段的需求不断增大。
  + 我国城镇化速度加快，城市面积不断增大，城市交通状况日益恶化，交通拥堵交通事敌频发，交通污染严重，使得城市必须借助智能交通监管手段来改变这样的现状和趋势。
  + 人们的生活水平的提高以及由于工作生活的需要，出行更加频繁.人们己不再满足汽车仅仅只是简单的代步工具，更希望它是安全的、舒适的、高效的和环保的，并带有娱乐、工作、辅助驾驶等功能，因而对定位导航、辅助驾驶、车载流媒体设备等车载电子设备提出了更高要求和迫切需求。
  + 我国智能交通的发展起步较晚，尚处在初级阶段，但发展较快，己经建设了很多系统并投入使用，如城市电子警察系统、城市红绿灯信号控制系统和城市交通状态检侧系统、城市交通信息实时发布系统、电子交通警察系统、电子不停车收费系统、高速公路紧急救援系统、高速公路智能运管系统、车载定位导航系统、车辆辅助驾驶系统等等。而以上系统相互独立，不能信息共享，需进行系统集成优化，傲到信息资源共享。
  + 我国交通状态复杂，行人、非机动车、机动车视行现象严重，导致交通违章行为和交通事故额繁发生，因此需要对行人和车辆给予足够的安全信息支持以保障其安全。

以上的挑战和机遇中，最先要解决的是车载环境下媒体的接入及控制。目前基于传统的无线网络媒体接入控制中，暂时没有专门针对车载环境下汽车高速移动的情况。基于以上原因，本文以车载环境下媒体接入控制作为研究对象，符合我们当前经济与社会发展的方向，并具有较好的现实意义。

车联网刚刚起步，还有许多挑战和问题，主要是RFID射频识别技术、传感技术、无线传输技术、云计算技术、车联网安全体系和定位技术等技术。以下是车联网在研究中必须重视的困难：

(1)频繁的链路断开和重连：高速行驶的车辆，导致了车辆图络拓扑结构的迅速变化，同时引起车间通信的间歇性中断。[13]

(2)高动态空时交通条件：不同行车区间的车辆密度有可能非常小的（例如，在农村地区）也有可能非常大（例如，城市上下班高峰期发生交通拥堵）。即使是在同一地点，一天之中车辆的密度也是动态变化的。解决高动态空时交通条件也是车联网络通信的难点之一，尤其是在车联网部署有初期阶段。[]

(3)数据传播的异质性：车联网络的设计必须考虑到能够同时有效支持道路安全信息和服务娱乐信息的应用。一般来说，道路安全信息相关应用需要低延时和高可靠性，相对的服务娱乐相关的应用也可保障合理的吞吐量，丢包率，资源利用的公平性。数据传播的异质性的决定了车联网信道接入和网络资源分配策略必须是自适应，从而达到高效、有序、公平的通信。[]

* 1. 选题的目的及意义

目前国内智能交通和车联网发展还有很大空间，也未形成一个完善的车联网产业链。虽然国内有部分高校和厂商在做这方面的研究，但投入到市场的产品有限，尚未规模化。但我们不能因为落后就不发展，相反应该迎头而上，大力研究和发展车联网。一方面社会的发展，必然会向智能交通、智能城市发展，另一方面民族崛起国家科技水平提升，不允许我国在车联网方面落后于国家其他国家和地区。总之，研究和发展车联网，有助于改善人民生活水平，促进社会发展，提高国家综合实力。

目前尚未有成熟的WAVE解决方案，因此我们这里在调查研究无线网络媒体接入控制技术和车联网相关技术的基础上，研究WAVE协议和IEEE802.11p协议，并在Linux操作系统编程和Linux内核网络子系统的基础上，借助Linux开放的网络协议栈接口，实现一个初步的双网卡媒体控制接入系统平台，具有非常重要的意义。除了研究CSMA/CA，这里还对TDMA的分配算法进行相关研究，具有前瞻性。

* 1. 论文主要研究内容及结构安排

本课题的主要工作是研究车载环境下的媒体接入控制，包括WAVE协议栈、双网卡工作方式以及TDMA时间片分配的相关研究。本文主要包括以下几部分，具体安排如下：

第一部分：绪论，阐述课题的研究背景、国内外车联网发展现状以及课题的研究目的和意义。

第二部分：相关技术和研究，首先是WAVE协议和IEEE1609协议族研究，包括WAVE协议和IEEE1609协议族架构、主要协议内容，然后讲述了Linux操作系统编程的相关内容。

第三部分：正文部分，主要按照IEEE1609协议族中特点，提出了双无线网卡工作方式的媒体控制接入系统平台，移植libpcap库。在Linux内核编程和驱动编程基础提出了802.11p驱动的实现流程。最后研究TDMA时间片分配算法，定义相关指标，通过仿真，验证了该算法的改进型。

第四部分：对全文进行总结及展望

第五部分：参考文献。

1. 相关技术
   1. 车联网相关技术概述

IEEE（美国电气和电子工程师协会）机构已经专门制定的协议标准WAVE（Wireless Access in Vehicular Environments）是由DSRC（Dedicated Short Range Communication）协议栈发展而来的，专门用于车联网中车-车、车-路的无线通信应用[14]。上世纪90年代不同国家根据各自国家情况制定不同的DSRC标准。美国ASTM（American Society for Testing and Materials）制定的DSRC标准时最早可见的，欧洲的DSRC标准则是由欧洲标准化委员会CEN（the European Committee for Standardization）制定，日本的DSRC标准由日本TC204委员会制定。中国则是参考了欧洲DSRC而制定了自己的标准。但是，这些标准均有待完善，许多问题善未解决，比如数据传输速率有待提高、路边无线终端设备覆盖范围有待加宽，需要进一步与互联网融合，因而应用有限，比如不停车收费（ETC）等[15]。因此，美国IEEE（美国电气和电子工程师协会）在DSRC标准基础上制定了WAVE协议（也即IEEE802.11p协议）。IEEE802.11p这个通讯协定主要用在车用电子的无线通讯上。它设计上是从IEEE802.11扩充延伸而来，来符合智慧型运输系统ITS的相关应用。

在IEEE802.11p协议基础上，IEEE又制定IEEE1609系列标准。IEEE1609协议系列是车载通信环境中针对无线电通信技术所定义的一系列的标准化服务接口和通讯系统架构。IEEE1609系列标准的主要目的是为了标准化车-车之间和车-路边基础设施之间的无线通信；同时为车载环境下的通信提供，包括交通安全、行车导航、自动收费以及交通管理等广泛应用所需要的通信标准。IEEE1609系列标准的制定考虑到了交通主管部门、汽车生产厂商、车辆服务提供商等各方面的生产管理需求。

IEEE1609系列协议族的体系如图二1所示：



图二1 IEEE1609协议族框架

从横向来看，整个框架分为两部分，一部分是管理平面Management Plane，另一部分是数据平面Data Plane。管理平面中包含管理信息库MIB，以及各种管理实体。管理信息指车载网中的网管框架中被管对象的集合。被管对象维护了可供管理程序读写的若干控制和状态信息。管理信息库代表某个车载终端设备或服务的一套可管理对象。数据平面是参考开放式互联系统（Open System Interconnection，OSI）[[[1]](#endnote-3)]七层网络模型来定义。最底层物理层PHY以及数据链路层中介质访问控制MAC子层下半部分由IEEE802.11p协议定义。数据链路层中介质访问控制MAC子层上半部分由IEEE1609.4协议定义，主要是定义了多信道协作。数据链路层中逻辑链路层（Logical Link Control，LLC）仍由IEEE802.2定义。再往上分为两种方式，一种是使用TCPv6/IPv6。另一种是由IEEE1609.3定义的WAVE短消息协议层（WAVE Short Message Protocol，WSMP），与传统的TCP/IP不同。安全应用和其他服务应用建立在它们的服务之上。最上面的IEEE1609.1则定义了WAVE的资源管理（Resource Manager）。除了上面这个图之外，还有一个没有出现在图中的IEEE1609.2协议，主要是定义了应用和管理消息的安全服务（Security Services for Applications and Management Messages）。

* 1. IEEE802.11p

IEEE802.11p协议是在IEEE802.11[[[2]](#endnote-4)]协议基础上针对车联网提出的修订。

* + 1. IEEE802.11协议

IEEE工作组制定了很多常见的协议，其中IEEE802规范定义了网卡如何访问传输介质（如光缆、双绞线、无线等），以及如何在传输介质上传输数据的方法，还定义了传输信息的网络设备之间连接建立、维护和拆除的途径。遵循IEEE802标准的产品包括网卡、桥接器、路由器以及其他一些用来建立局域网络的组件[[[3]](#endnote-5)]。IEEE802委员会分为三部分：传输介质分会，研究局域网物理层协议；信号访问控制分会，研究数据链路层协议；高层接口分会，研究网络层到应用层的有关协议。

IEEE802是一个局域网标准系列，常见的协议有：

* IEEE802.1A------局域网体系结构
* IEEE802.1d------生成树协议Spanning Tree
* IEEE802.1p------General Registration Protocol
* IEEE802.1q------虚拟局域网Virtual LANs：VLan
* IEEE802.1w------快速生成树协议 RSTP
* IEEE802.1s------多生成树协议MSTP
* IEEE802.1x------基于端口的访问控制Port Based Network Access Control
* IEEE802.1g------Remote MAC Bridging
* IEEE802.1v------VLAN Classification by Protocol and Port[1]
* IEEE802.1B------寻址、网络互连与网络管理
* IEEE802.2-------逻辑链路控制(LLC)
* IEEE802.3-------CSMA/CD访问控制方法与物理层规范
* IEEE802.3i------10Base-T访问控制方法与物理层规范
* IEEE802.3u------100Base-T访问控制方法与物理层规范
* IEEE802.3ab-----1000Base-T访问控制方法与物理层规范
* IEEE802.3x------是全双工以太网数据链路层的流控方法。当客户终端向服务器发出请求后，自身系统或网络产生拥塞时，它会向服务器发出PAUSE帧，以延缓服务器向客户终端的数据传输。
* IEEE802.3z------1000Base-SX和1000Base-LX访问控制方法与物理层规范
* IEEE802.4-------Token-Bus访问控制方法与物理层规范
* IEEE802.5-------Token-Ring访问控制方法
* IEEE802.6-------城域网访问控制方法与物理层规范
* IEEE802.7-------宽带局域网访问控制方法与物理层规范
* IEEE802.8-------FDDI访问控制方法与物理层规范
* IEEE802.9-------综合数据话音网络
* IEEE802.10------网络安全与保密
* IEEE802.11------无线局域网访问控制方法与物理层规范
* IEEE802.12------100VG-AnyLAN访问控制方法与物理层规范
* IEEE802.14------协调混合光纤同轴(HFC)网络的前端和用户站点间数据通信的协议。
* IEEE802.15------无线个人网技术标准，其代表技术是zigbee。
* IEEE802.16------宽带无线MAN标准－WiMAX
* IEEE802.17------弹性分组环（RPR）工作组
* IEEE802.18------宽带无线局域网技术咨询组（Radio Regulatory）
* IEEE802.19------多重虚拟局域网共存技术咨询组
* IEEE802.20------移动宽带无线接入（MBWA）工作组

IEEE802.11中专门定义了无线局域网访问控制方法与物理层规范，这也是一个系列，包含了不同的协议，如表二1所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 协议 | 发布日期 | 频带 | 有效传输距离 | 最大传输速度 |
| 802.11 | 1997 | 2.4-2.5GHz | 100米 | 2Mbit/s |
| 802.11a | 1999 | 5.15-5.35/5.47-5.725/5.725-5.875GHz | 75米 | 54Mbit/s |
| 802.11b | 1999 | 2.4-2.5GHz | 100米 | 11Mbit/s |
| 802.11g | 2003 | 2.4-2.5GHz | 150米 | 54Mbit/s |
| 802.11n | 2009 | 2.4GHz或者5GHz | 150米 | 600Mbit/s |
| 802.11p | 2010 | 5.850-5.925GHz | 1公里[[[4]](#endnote-6)] | 27Mbit/s |

表二1 IEEE802.11协议族主要通信协议列表

IEEE802.11无线协议同其他的IEEE802标准一样，主要工作在ISO七层协议的最底两层，物理层PHY和数据链路层LNK。由于IEEE802.2逻辑链路控制层LLC的存在，对上层透明化下层的差异，所以，任何局域网的应用程序、网络操作系统都能运行在IEEE802.11上。

IEEE802.11协议定义了两类通信设备：工作基站（Access Point，AP）和无线工作站（Station，STA）。[[[5]](#endnote-7)]AP一般由一个无线口和有线口构成，一方面作为网络接入点负责控制无线网络，另一方面实现无线网络与有线网络之间的桥接，将STA联通到有线网络，从而接入到Internet。IEEE802.11协议定义了两种工作模式：基本服务集合模式（Basic Service Set，BSS）和独立基本服务集合模式（Independent Basic Service Set，IBSS）。BSS中，至少存在一个中心控制节点AP，负责中心控制、调度和认证等，其他的无线终端节点STA均连到这个AP中，即使两台STA之间要通信，也是通过AP来控制转发。两个或者两个以上的BSS可构成扩展服务集合（Extended Service Set，ESS），比如候机场、图书馆等，用户看到是单独的一个BSSID，而实际背后是由多个AP互联组成的。IBSS中，不存在中心控制节点AP，仅由无线终端节点组成。IBSS也即是ad hoc。IBSS中，一般根据某种算法来选择STA作为中心控制节点。IBSS主要目的是用户自发地形成一个无线局域网，用于分享文件等，这种网络一般都是临时性的。

在有线网络中比如以太网token网中，由于需要通过实体物理线路接入到中心控制设备的接口，才能加入相应的网络，而且中心控制设备一般集中存放在机房中，所以要控制终端设备是否要接入网络中，操作上相对简单，铺设线路到允许接入的地方。对于终端设备来说，它的一个网口在同一时刻只能接入到一个网络中，要切换网络，必须切换实体物理线路。而无线网络中，它的媒介是无线电磁波，并不像实体物理线路那样具有定向性和可控制性。如果不加任何控制，那么处于无线覆盖范围内的任意一个无线终端节点均能接入到无线网络中。同时，对于无线终端设备中，它往往能监听到多个无线网络存在。所以需要在软件层次上来控制无线网络的接入。在BSS和IBSS中，STA加入无线网络均得经过关联（Association）和认证（Authentication）过程。

一般中心节点会周期性地广播它的BSSID，所以STA最开始先扫描可以使用的网络。如果中心节点不广播它的BSSID，那这种方式下使用者必须通过其他方式来获得BSSID，比如人与人口头之间交流，这种方式更加限制陌生用户的使用。然后STA针对某个无线网络进行关联，关联即为STA绑定到这个无线网络上来，忽视其他的无线网络，未关联的STA，相当于有线网络中线路上没有连接到中心控制设备上。关联之后，STA就开始在针对这个无线网络收发数据。对于开放使用的网络，对用户没有限制，这时STA无需认证过程就可以在该无线网络中自由使用了。对于有限制的网络，还需要认证过程，限制符合身份的STA使用无线网络，由于无线媒介对所有STA可见，所有STA都能收到无线网络中的数据报，所以认证过程不传输明文，通常是传输经过加解密的认证数据，比如WEP Open System、WEP Share Key、WPA-PSK、WPA2-PSK、WPA-EAP和WPA2-EAP等。还可以更高层次的认证过程，比如限定MAC，网页认证等，这些不属于强制内容，不在这里讨论。

无线网络中，信号可能有意或无意被干扰。一个终端节点发送数据之后，有可能这个数据会丢失，对方可能接收不到。因此大部分传输需要采用肯定确认机制（Positive acknowledge of data transmissions）。这种机制要求发送一个Frame给对方时，对方必须回应一个Ack确认帧给源节点。源节点只有在收到了Ack帧才能算该Frame发送成功。否则，需要重发该帧或者重发次数过多而丢弃该帧。如图二2所示：



图二2 肯定确认机制

STA在加入网络后，一般不止一个STA。在网络通信过程中，这些STA使用同一个媒介同一信道。无论有线还是无线，同一时刻不能有两个或两个以上同时利用媒介发送数据，否则信号会互相干扰。STA需要先监听信道是否正在占用，是否忙，忙就退避一段时间。IEEE802.11中使用了物理监听和虚拟监听。物理监听就是监听当前无线电媒介是否忙。忙则自身不能发送数据。虚拟监听是一种定时器，网络分配矢量（Network Allocation Vector，NAV）。如图二3所示：



图二3 网络分配矢量NAV

如图所示，发送者在发送RTS、Frame帧时，均有相应帧回应，所以其他的STA中标记信道忙的时间应该直到ACK帧发送结束。这时从逻辑上设定的虚拟监听。

无线网络中有两个问题值得注意，一个是隐藏节点（hidden nodes），一个是暴露节点（exposed nodes）。

隐藏节点是指在一个节点处于接收者的通信范围内而在发送者的通信范围外。该节点因为监测不到发送节点的无线电信号，导致和发送节点同时向目标接收节点发送报文，这样子在接收节点出会造成冲突。如图二4所示，节点A能感知节点B的存在，节点B能感知到节点A、C的存在，节点C能感知到节点B的存在。但A、C两个节点无法感知到对方存在。如果节点A、C同时向节点B发送数据，那么节点B收到的是两个信号的混扰，这个过程是错误的。解决这个问题方法之一是使用请求发送/允许发送（Request To Send/Clear To Send，RTS/CTS）。节点A或者节点C必须先向B节点发送一个RTS帧，等待B发送CTS帧之后才能节点A或者节点C才能发送数据。因为节点B发送的CTS都能被节点A和C收到，所以节点A或者C是在告诉对方自己有数据要发送。如图二5所示：



图二4 隐藏节点



图二5 使用RTS/CTS



图二6 暴露节点

暴露节点是指终端处于发送者的通信范围内而在接收者的通信范围之外。节点因为检测到发送节点的发送信号而延迟发送，但实际因为它处于接收节点的通信范围之外，所以它的发送实际上不会造成冲突。由于推迟处理，这样子会导致网路容量降低。如图二6所示，节点B向A发送数据，C也在向节点D发送数据。由于节点B、C能感知到对方的存在，所以一方在发送数据时，另一方必须等待，延迟一段时间才能发送。这种问题即为暴露节点问题，会增加不必要的延迟，特别是车载环境中实时性非常重要的应用。而实际逻辑上，节点B发给节点A的时候，C节点也可以发给节点D，虽然节点B、C出信号会混淆，但节点A、D最终都能收到正确的信号。

IEEE802.11定义了两种MAC控制方法：分布式协调功能（Distributed Coordination Function，DFC）和点协调功能（Point Coordination Function，PCF）。DCF是节点共享无线信道进行数据传输的基本接入方式，它把CSMA/CA（Carrier Sense Multiple Access with Collision Avoidance）技术和确认（ACK）技术结合起来。它采用二进制指数回退策略来避免冲撞。PCF是一种可选的无线网络媒介访问控制方法，主要用于支持近乎实时的应用。

这些方法并没有百分百地确保冲突不发生，而是尽可能减少冲突发生。当无线网络中终端节点数目越来越多时，冲突几率也会越来越大。

* + 1. IEEE802.11p

802.11p是作为IEEE802.11的修正议而出现，遵循相同的IEEE802.11协议架构，目的是为了适应车联网环境。在PHY上，802.11p采用了OFDM[[[6]](#endnote-8)]，与802.11a一致，这种方式抗干扰性强，更能适合快速变化的车辆环境，信道带宽缩小一半，从IEEE802.11a的20MHz变成10MHz，从而减小多普勒效应，同时也减少了码间串扰。IEEE802.11p还将信道进行划分，将5.850-5.925GHz这75MHz的带宽划分为7个channel。



图二7 WAVE频谱分配图

如图二7所示，IEEE802.11p定义了两种类型信道，一种是控制信道（Control Channel，CCH），另一种是服务信道（Service Channel，SCH）。Ch178即为CCH，频率范围为5.885GHz-5.895GHz，主要负责传输实时性要求比较高的数据，以及WSA报文，Ch174、Ch176、Ch180和Ch182都是SCH，主要负责传输实时性要求不高的应用或者娱乐商务应用，如视频通话、邮件等应用。最左Ch172和最右Ch184的信道保留着专门用途。5.850-5.855GHz这频带为空闲频带，不使用。

IEEE802.11p中，取消了AP（Access Point）和STA（Station）的概念，将网络设备分为RSU（Road-Side Unit）和OBU（On-Board Unit）两类[[[7]](#endnote-9)]。其中，RSU所扮演角色即是AP，OBU设备属于车载设备。RSU一般安装固定在道路两侧，并利用有线或无线的方式将其与互联网联通。WAVE网络组网可以有两种方式：WBSS（WAVE Basic Service Set）和WIBSS（WAVE Independent Basic Service Set）。WBSS是在RSU存在的情况下，一个RSU和多个OBU组成的网络；WIBSS是在缺失RSU的情况下，多个OBU之间组成的网络，此网络归属于移动的Ad-hoc类型的网络。RSU和OBU之间基于WAVE协议栈定义的报文格式进行交互通信。CCH上会周期性地广播WSA（WAVE Service Announcement），其作用相当于IEEE802.11网络中Beacon报文作用。OBU在进入WBSS覆盖范围中，通过收到的WSA选择加入WBSS，离开WBSS覆盖范围后，将退出此WBSS网络。

IEEE802.11p协议还新增了一种“MAVE Mode”模式，在这种模式下，OBU加入无线网络时，没有信道扫描、关联和认证等加入网络的过程，直接在无线网络中发起通信，减少冗长的认证过程，适应了车载环境下车辆快速移动从而停留在无想网络覆盖范围内时间大大缩短的情况。而非WAVE Mode模式下，其加入网络过程为：

OBU发送probe request->

<-RSU发送probe response

OBU发送身份验证Authentication request->

<-RSU发送Authentication Confirm

OBU发送关联Association request->

<-RSU发送Association Confirm，注册一个用户

* 1. IEEE1609协议栈

IEEE1609标准是以IEEE802.11p通讯协定为基础的高层标准，是车联网协议的组成部分，是IEEE针对车载环境所定义的通讯系统架构和标准化服务接口。IEEE1609协议族在11p协议基础上，定义上层相关的协议，包括数据链路层、IP网络层、数据传输层、通信安全等方面，定义了多信道协调工作来增强信道可用性，区分紧急重要数据和普通数据。制定IEEE1609系列标准的主要目的是为了标准化车-车之间和车-基础设施之间的无线通信；同时为车载环境下的通信提供，包括交通安全、行车导航、自动收费以及交通管理等等广泛应用所需要的通信标准[[[8]](#endnote-10)]。IEEE1609系统列标准协议如表二2所示：

|  |  |  |
| --- | --- | --- |
| 协议 | 发布日期 | 用途 |
| IEEE1609.0 | 2012年  草案 | 总体描述了适用于车载环境下的WAVE/DSRC架构和服务，并介绍说明了其它IEEE1609系列协议标准的作用和目的[[[9]](#endnote-11)]。 |
| IEEE1609.1 | 2006年  草案 | 定义了WAVE架构的资源管理功能，也即是各协议层及应用层之间的介质管理，另外定义WAVE设备具有额外的管理机制，使具有控制能力的节点能够远端控制一个区域内的所有节点[[[10]](#endnote-12)]。 |
| IEEE1609.2 | 2013年 | 定义了WAVE架构的安全机制。 |
| IEEE1609.3 | 2010年 | 定义了WAVE系统中的网路层通讯协定及管理机制[[[11]](#endnote-13)]。主要内容是制定了短距离通信协议（WSMP），WAVE设备之间数据传输时间能够有效的降低了；定义了管理实体层（WME），应用程序需要通过WAVE协议栈进行通信时需要向WME进行注册。 |
| IEEE1609.4 | 2011年 | 定义了WAVE系统中MAC层的多信道协调机制[[[12]](#endnote-14)]。WAVE系统中包含一个控制信道CCH与多个服务信道SCH，不同的WAVE设备之间必须利用统一的控制信道进行安全和管理信息的交换，并使用不同的服务信道进行其它业务数据的通信。因此WAVE设备必须能够定期或不定期的在控制信道和服务信道之间进行切换。 |
| IEEE1609.5 | 2011年  草案 | 定义了WAVE系统中车-车以及车-路边基站之间的无线连接接入和通信管理服务。 |
| IEEE1609.11 | 2011年 | 主要针对WAVE系统中的电子收费应用。其主要内容为制定以WAVE设备为基础之电子收费系统标准。 |
| IEEE1609.12 | 2012年  草案 | 在WAVE协议栈中，尤其是在IEEE1609系列标准中使用了许多标识符，IEEE1609.12协议即描述了这些标识符的分配和使用。 |

表二2 IEEE1609系列

其中，IEEE1609.4的多信道协调，是针对IEEE802.11p协议的扩展。多信道协调方式指信道划分为控制信道CCH和服务信道SCH，控制信道负责传输对实时性要求较高的报文以及传输协议控制报文，比如安全相关的车辆故障紧急通知、车辆防碰撞预警信息，以及传输管理控制相关的服务信息广播等；SCH传输实时性和可靠性要求不高的数据，比如道路阻塞、最优路径、天气预报、娱乐新闻等[[[13]](#endnote-15)]。由于单个天线在某个时刻只能以一个频率收发无线信号，所以要同时支持两种信道，要么需要两根以上天线，要么单根天线不断地切换频道频率。IEEE1609.4中明确了单天线的信道协作机制。不同的无线设备之间的这种信道协作机制，要求通信对象之间时间上的同步，使得任意时刻无线设备工作在约定的信道上。如图二 8 所示。



图二8 CCH和SCH的间隔

IEEE1609.4协议中允许使用GPS卫星信号中的时间信息来同步所有设备，也允许RSU广播WSA广播信号进行同步。多天线多无线模块，可选择其中一个工作在控制信道上，剩下的在服务信道上工作。



图二9 信道协调机制中的两个MAC实体

IEEE1609.4协议所定义的多信道协调机制框架建立在802.11p所定义的MAC实体之上，SCH和CCH均看成独立存在的MAC。如图二9 所示，MAC中新增了一个channel Router，用来路由上层报文，选择控制信道CCH还是服务信道SCH。对于实时性要求高的报文只能路由到控制信道CCH上，其他的普通数据则路由到服务信道SCH。上。控制信道和服务信道进一步采用了EDCA机制，有4个发送队列，不同发送队列有不同的优先级。一个上层数据报发送出去的流程如下：MAC收到上层的数据包之后，首先经过Channel Router路由选择合适的信道，然后根据上层数据包的发送优先级选择合适的发送队列，接着EDCA机制根据设置选择一个上层数据包传递给下层的MAC实体，最后改MAC实体通过DCF竞争取得物理无线媒介发送权时将数据发送出去。可以看到，数据在发送过程中会经过多个层次的竞争，形成不同等级的优先权，紧急的数据优先级比较高，会被尽快发送出去，延迟小，而一般的普通数据则优先级较低，会有稍高的延迟，从而保证WAVE的QoS。由于WAVE多了Channel Router等，所以上层数据包的头部需要新增一些参数配置，比如功率、信道编号、数据速率等。

IEEE1609.3协议定义了WAVE的无线接入网络层服务WAVE短消息协议（WAVE Short Message Protocol，WSMP），对应OSI模型的网络层和传输层，主要提供报文传输和寻址。IEEE1609.3还定义了用于网路管理的WAVE管理实体（WAVE Management Entity，WME）。当WAVE设备的通信发起方想要发送的数据时，其设定通信目的MAC地址为广播地址，设定WSMP参数，经过一定步骤之后在信道上发送出去。目的WAVE设备收到WSMP帧之后，根据其中携带的Provider Service ID来将WSM转发给对应的上层服务程序。WSMP可在CCH和SCH上传输。IEEE1609.3定义了WAVE系统中设备的两种角色，一种是服务提供者（Provider），一种是服务使用者（User）。建立服务过程中，服务提供者在CCH上广播WSA，通知其他设备有哪些服务内容。WSA包括了服务提供者的工作信道、传输速率、服务ID、优先级等，由WME产生。User在控制信道上监听WSA，搜集形成一个有效的服务列表。当User的上层应用程序需要使用某个服务时，就从服务列表中找到所需服务的信息，然后根据该信息所包含的工作信道、传输速率、服务ID、优先级等参数，在适当的时间跳转到适当的频道上以适当的速率接收服务提供者所提供的服务。对于实时性较高的数据应用，可以选择直接在CCH上传输，这种方式不需要WSA广播，直接对等通信，进一步减小延迟。

* 1. Linux相关编程

常见的嵌入式系统大体上分为两类，商用型和免费性。

* + 商用型的实操作系统功能稳定、可靠，有完善的技术支持和售后服务，但往往价格昂贵。
  + 免费型的实时操作系统在价格方面具有优势，目前主要有Linux和uC/OS，稳定性与服务性存在挑战。

以下是常见的嵌入式操作系统[[[14]](#endnote-16)]：

* + VxWorks操作系统是美国WindRiver公司于1983年设计开发的一种嵌入式实时操作系统（RTOS），具有良好的持续发展能力、高性能的内核以及友好的用户开发环境，在嵌入式实时操作系统领域牢牢占据着一席之地。主要特点是可靠性、实时性和可裁剪性，支持多种处理器。
  + Windows Embedded是一种针对小容量、移动式、智能化、连接设备的模块化实时嵌入式操作系统。它是针对掌上设备、无线设备的动态应用程序和服务提供了一种功能丰富的操作系统平台，WindowsCE嵌入但不够实时，属于软实时操作系统，由于其Windows背景，界面比较统一认可。
  + Palm OS是著名的网络设备制造商3COM旗下的Palm Computing掌上电脑公司的产品。Palm OS在PDA市场上占有很大的市场份额，Palm OS的市场份额占到将近90%，最近下降70％，目前主要与WIN CE进行激烈竞争。
  + 嵌入式Linux的内核小、功能强大、运行稳定、系统健壮、效率高，易于定制剪裁，在价格上极具竞争力。

Linux是一套免费使用和自由传播的类Unix操作系统，是一个基于POSIX和UNIX的多用户、多任务、支持多线程和多CPU的操作系统，具有BSD和SYSV的扩展性。它能运行主要的UNIX工具软件、应用程序和网络协议。Linux继承了Unix以网络为核心的设计思想，是一个性能稳定的多用户网络操作系统。

Linux操作系统诞生于1991年的10月5日。Linux存在着许多不同的Linux版本，但它们都使用了Linux内核。Linux可安装在各种计算机硬件设备中，比如手机、平板电脑、路由器、视频游戏控制台、台式计算机、大型机和超级计算机。在移动设备上广泛使用的Android操作系统就是创建在Linux内核之上。Linux也是自由软件和开放源代码软件发展中最著名的例子。只要遵循GNU通用公共许可证，任何个人和机构都可以自由地使用Linux的所有底层源代码，也可以自由地修改和再发布。严格来讲，术语Linux只表示操作系统内核本身，但通常采用Linux内核来表达该意思。Linux则常用来指基于Linux内核的完整操作系统，包括GUI组件和许多其他实用工具。由于这些支持用户空间的系统工具和库主要由理查德·斯托曼于1983年发起的GNU计划提供，自由软件基金会提议将该组合系统命名为GNU/Linux，但Linux不属于GNU计划。

Linux最初是作为支持英特尔x86架构的个人电脑的一个自由操作系统。目前Linux已经被移植到更多的计算机硬件平台，远远超出其他任何操作系统。Linux是一个领先的操作系统，可以运行在服务器和其他大型平台之上，如大型主机和超级计算机。

Linux的基本思想有两点：第一，一切都是文件；第二，每个软件都有确定的用途。其中第一条详细来讲就是系统中的所有都归结为一个文件，包括命令、硬件和软件设备、操作系统、进程等等对于操作系统内核而言，都被视为拥有各自特性或类型的文件。至于说Linux是基于Unix的，很大程度上也是因为这两者的基本思想十分相近。

* + 1. Linux的特点
* 完全免费

Linux是一款免费的操作系统，用户可以通过网络或其他途径免费获得，并可以任意修改其源代码。这是其他的操作系统所做不到的。正是由于这一点，来自全世界的无数程序员参与了Linux的修改、编写工作，程序员可以根据自己的兴趣和灵感对其进行改变，这让Linux吸收了无数程序员的精华，不断壮大。

* 开放性

开放性是指系统遵循世界标准规范，特别是遵循开放系统互连（OSI）国际标准。凡遵循国际标准所开发的硬件和软件，都能彼此兼容，可方便地实现互连。

* 完全兼容POSIX1.0标准

这使得可以在Linux下通过相应的模拟器运行常见的DOS、Windows的程序。这为用户从Windows转到Linux奠定了基础。许多用户在考虑使用Linux时，就想到以前在Windows下常见的程序是否能正常运行，这一点就消除了他们的疑虑。

* 多用户

多用户是指系统资源可以被不同用户各自拥有使用，即每个用户对自己的资源（例如：文件、设备）有特定的权限，互不影响。Linux和Unix都具有多用户的特性

* 多任务

多任务是现代计算机的最主要的一个特点。它是指计算机同时执行多个程序，而且各个程序的运行互相独。Linux系统调度每一个进程平等地访问微处理器。由于CPU的处理速度非常快，其结果是，启动的应用程序看起来好像在并行运行。事实上，从处理器执行一个应用程序中的一组指令到Linux调度微处理器再次运行这个程序之间只有很短的时间延迟，用户是感觉不出来的。

* 良好的界面

Linux向用户提供了两种界面：用户界面和系统调用。

Linux的传统用户界面是基于文本的命令行界面，即shell，它既可以联机使用，又可存在文件上脱机使用。shell有很强的程序设计能力，用户可方便地用它编制程序，从而为用户扩充系统功能提供了更高级的手段。可编程Shell是指将多条命令组合在一起，形成一个Shell程序，这个程序可以单独运行，也可以与其他程序同时运行。

Linux还为用户提供了图形用户界面。它利用鼠标、菜单、窗口、滚动条等设施，给用户呈现一个直观、易操作、交互性强的友好的图形化界面。

系统调用给用户提供编程时使用的界面。用户可以在编程时直接使用系统提供的系统调用命令。系统通过这个界面为用户程序提供低级、高效率的服务。

* Linux的设备独立性

设备独立性是指操作系统把所有外部设备统一当作成文件来看待，只要安装它们的驱动程序，任何用户都可以象使用文件一样，操纵、使用这些设备，而不必知道它们的具体存在形式。Linux是具有设备独立性的操作系统，它的内核具有高度适应能力，随着更多的程序员加入Linux编程，会有更多硬件设备加入到各种Linux内核和发行版本中。另外，由于用户可以免费得到Linux的内核源代码，因此，用户可以修改内核源代码，以便适应新增加的外部设备。

* Linux供了丰富的网络功能

完善的内置网络是Linux的一大特点。Linux在通信和网络功能方面优于其他操作系统。其他操作系统不包含如此紧密地和内核结合在一起的连接网络的能力，也没有内置这些联网特性的灵活性。支持Internet:Linux免费提供了大量支持Internet的软件，Internet是在Unix领域中建立并繁荣起来的，在这方面使用Linux是相当方便的，用户能用Linux与世界上的其他人通过Internet网络进行通信。文件传输:通过Linux命令完成内部信息或文件的传输。远程访问:Linux为系统管理员和技术人员提供了访问其他系统的窗口。通过这种远程访问的功能，一位技术人员能够有效地为多个系统服务，即使那些系统位于相距很远的地方。

* Linux提供了可靠的系统安全

Linux采取了许多安全技术措施，包括对读、写进行权限控制、带保护的子系统、审计跟踪、核心授权等，这为网络多用户环境中的用户提供了必要的安全保障。

* Linux支持多种平台

Linux可以运行在多种硬件平台上，如具有x86、680x0、SPARC、Alpha等处理器的平台。此外Linux还是一种嵌入式操作系统，可以运行在掌上电脑、机顶盒或游戏机上。2001年1月份发布的Linux2.4版内核已经能够完全支持Intel64位芯片架构。同时Linux也支持多处理器技术。多个处理器同时工作，使系统性能大大提高。

Linux相对比较不耗系统资源

Linux只要p-100以上等级的处理器就可以安装并且使用愉快，还不需要到P-III等级的嵌入式系统呢！不过，如果你要架设的是属于大型的主机（服务上百人以上的主机系统），那么就需要比较好一点的机器了。不过，目前市面上任何一款个人计算机均可以达到这一个要求。

* + 1. Linux体系结构

Linux操作系统由4个主要的子系统所组成[[[15]](#endnote-17)]：

* 用户应用程序—在某个特定的Linux系统上运行的应用程序集合，它将随着该计算机系统的用途不同而有所变化，但一般会包括文字处理应用程序和Web浏览器
* O/S服务—这些服务一般认为是操作系统的一部分（XWindow系统，命令外壳程序shell等等）；此外，内核的编程接口（编译工具和库）也属于这个子系统。
* Linux内核—包括内核抽象和对硬件资源（如CPU）的间接访问。
* 硬件控制器—这个子系统包含在Linux实现中所有可能的物理设备，例如，CPU、内存硬件、硬盘以及网络硬件等都是这个系统的成员。

Linux内核由5个主要的子系统构成

* 进程调度程序（SCHED）负责控制进程访问CPU。调度程序所使用的策略可以保证进程能够公平地访问CPU，同时保证内核可以准时执行一些必需的硬件操作。
* 内存管理程序（MM）使多个进程可以安全地共享机器的主存系统。此外，内核管理程序支持虚拟内存。虚拟内存使得Linux可以支持进程使用超过系统中的内存数量的内存。暂时用不着的存储信息可以交换出内存，存放到使用文件系统的永久性存储器上，然后在需要它们的时候再交换回来。
* 虚拟文件系统（VFS）。通过提供一个所有设备的公共文件接口，VFS抽象了不同硬件设备的细节。此外，VFS支持与其他操作系统兼容的不同的文件系统格式。
* 网络接口（NET）提供了对许多建网标准和网络硬件的访问。
* 进程间通信（IPC）子系统。

Linux与其他操作系统的区别有：

（1）Linux与主要UNIX系统

* Linux采用了SVR4的进程间通信（IPC）机制：共享内存、消息队列、信号灯
* Linux支持BSDSocket网络编程接口
* 许多Linux发行版采用SysV init机制，支持运行级别
* LINUX具有Unix的全部功能，任何使用Unix操作系统或想要学习Unix操作系统的人都可以从Linux中获益

（2）Linux与MS-DOS

* MS-DOS没有完全实现x86处理器的功能，而Linux完全在处理器保护模式下运行，并且开发了处理器的所有特性。
* Linux可以直接访问计算机内的所有可用内存，提供完整的Unix接口。而MSDOS只支持部分Unix的接口。
* MS-DOS是单任务的操作系统

(3)Linux与OS/2、Windows、Windows NT

Linux是从一个比较成熟的操作系统发展而来的，而其他操作系统，如Windows NT等，都是自成体系，无对应的相依托的操作系统。（Linux是Unix的一个克隆）

Linux是一种开放、免费的操作系统，而其他操作系统都是封闭的系统，需要有偿使用。

* + 1. Linux设备驱动程序

操作系统的主要作用有两个，一个是屏蔽底层千差万别的硬件差别，给上层提供一个相同的接口，另外一个是管理和调度系统资源。驱动程序主要是透明化硬件的差别，使得用户无需关心硬件的差别。在Linux中，所有设备均抽象化文件，所以所有用户在使用硬件时跟使用普通文件并无多大差别。设备驱动处于物理硬件和内核上层之间，具有重要的桥梁作用。Linux中设备分为三大类，字符设备、块设备和网络设备[[[16]](#endnote-18)]。

* 字符设备

字符（char）设备是个能够像字节流（类似文件）一样被访问的设备，由字符设备驱动程序来实现这种特性。字符设备驱动程序通常至少要实现open、close、read和write的系统调用。字符终端（/dev/console）和串口（/dev/ttyS0以及类似设备）就是两个字符设备，它们能很好的说明“流”这种抽象概念。字符设备可以通过FS节点来访问，比如/dev/tty1和/dev/lp0等。这些设备文件和普通文件之间的唯一差别在于对普通文件的访问可以前后移动访问位置，而大多数字符设备是一个只能顺序访问的数据通道。然而，也存在具有数据区特性的字符设备，访问它们时可前后移动访问位置。例如framebuffer就是这样的一个设备，app可以用mmap或lseek访问抓取的整个图像。

* 块设备

和字符设备类似，块设备也是通过/dev目录下的文件系统节点来访问。块设备（例如磁盘）上能够容纳filesystem。在大多数的Unix系统中，进行I/O操作时块设备每次只能传输一个或多个完整的块，而每块包含512字节（或2的更高次幂字节的数据）。Linux可以让app像字符设备一样地读写块设备，允许一次传递任意多字节的数据。因此，块设备和字符设备的区别仅仅在于内核内部管理数据的方式，也就是内核及驱动程序之间的软件接口，而这些不同对用户来讲是透明的。在内核中，和字符驱动程序相比，块驱动程序具有完全不同的接口。

* 网络设备

任何网络事物都需要经过一个网络接口形成，网络接口是一个能够和其他主机交换数据的设备。接口通常是一个硬件设备，但也可能是个纯软件设备，比如回环（loopback）接口。网络接口由内核中的网络子系统驱动，负责发送和接收数据包。许多网络连接（尤其是使用TCP协议的连接）是面向流的，但网络设备却围绕数据包的传送和接收而设计。网络驱动程序不需要知道各个连接的相关信息，它只要处理数据包即可。由于不是面向流的设备，因此将网络接口映射到filesystem中的节点（比如/dev/tty1）比较困难。Unix访问网络接口的方法仍然是给它们分配一个唯一的名字（比如eth0），但这个名字在filesystem中不存在对应的节点。内核和网络设备驱动程序间的通信，完全不同于内核和字符以及块驱动程序之间的通信，内核调用一套和数据包相关的函数而不是read、write等。

Linux内核编程中，和用户态编程有很大的区别。最主要是内核模块运行于内核空间，具有很大地权限，可以访问所有的硬件资源，一旦出错，系统可能直接崩溃宕机。而应用程序运行用户态中，所有资源都是内核调度提供的，程序一旦出错，一般只有该程序会结束，而系统仍照常运行。由于内核模块运行在内核态中，所以并发控制、中断、内存映射等都需额外小心。

* 1. 本章小结

本章主要介绍了论文的相关技术背景，首先提出了IEEE1609框架，这是整个车联网协议框架的核心，分为数据平面和管理平面两大块。接着讨论了IEEE802.11无线协议，定义了物理层和数据链路层的标准，同时也提出了AP/STA、BSS/IBSS等概念，介绍了隐藏节点、暴露节点等问题。之后介绍了IEEE802.11p，这个协议主要是IEEE802.11协议针对车联网所做出的修订，提出信道划分、WAVE模式。IEEE1609.4提出了多信道协调机制，IEEE1609.3提出了无线接入网络层服务WAVE短消息协议。本章最后主要是介绍了Linux操作系统的特点，Linux操作系统的体系架构，Linux内核的结构，Linux设备驱动。Linux是一个开放、自由的操作系统，适用于各种平台和应用。

1. 基于双网卡的媒体接入控制优化
   1. 方案描述

802.11p和IEEE1609协议中规定了控制信道和服务信道两种信道，控制信道主要是用来传输重要紧急的数据，而服务信道用来传输普通的应用数据。这里首先提出利用目前成熟的IEEE802.11协议网卡双无线网卡通信平台系统方案来模拟WAVE中的两种信道方式。本设计中，在应用层使用libpcap函数库直接指定虚拟网卡进行收发数据，然后在这基础上实现控制信道和服务信道工作方式。



图三1 车联网通信终端系统硬件架构

硬件方案开发平台选用友善之臂的Tiny6410开发板，无线网卡选RT3070无线网卡。



图 三2 tiny6410实物图

Tiny6410是一款以ARM11芯片（三星S3C6410）作为主处理器的嵌入式核心板，该CPU基于ARM1176JZF-S核设计，内部集成了强大的多媒体处理单元，支持Mpeg4，H.264/H.263等格式的视频文件硬件编解码，可同时输出至LCD和TV显示；它还并带有3D图形硬件加速器，以实现OpenGL ES1.1&2.0加速渲染，另外它还支持2D图形图像的平滑缩放，翻转等操作。Tiny6410采用高密度6层板设计，尺寸为64x50mm，它集成了256M Mobile DDR RAM，256M/1GB SLC Nand Flash存储器，采用5V供电，在板实现CPU必需的各种核心电压转换，还带有专业复位芯片，通过2.0mm间距的排针，引出各种常见的接口资源，以供不打算自行设计CPU板的开发者进行快捷的二次开发使用。Tiny6410SDK是采用Tiny6410核心板的一款参考设计底板，它主要帮助开发者以此为参考进行核心板的功能验证以及扩展开发。该底板具有三LCD接口、4线电阻触摸屏接口、100M标准网络接口、标准DB9五线串口、Mini USB2.0接口、USB Host 1.1、3.5mm音频输入输出口、标准TV-OUT接口、SD卡座等常用接口；另外还引出4路TTL串口，另1路TV-OUT、SDIO2接口（可接SD WiFi）接口等；在板的还有蜂鸣器、I2C-EEPROM、备份电池、AD可调电阻、8个中断式按键等。

本硬件平台上的操作系统为Linux操作系统，源码开放，移植性好，可裁剪和定制性高。开发板同时提供了相应配套的开发套件，比如编译器arm-linux-gcc等。

每个终端系统均由tiny6410和两块无线网卡组成。这两个网卡在逻辑上是绑定成一个虚拟网卡提供给上层。系统应用程序在这个虚拟网卡进行传输。在这次实现中，虚拟网卡主要是虚拟成一个进程方式提供给其他进程。

按照IEEE802.11p和IEEE1609协议族的定义，信道分为控制信道和服务信道，因此设置无线网卡A一直工作在控制信道CCH上，无线网卡B一直工作在服务信道SCH上，如图二7 。无线网卡A主要用于传输紧急重要信息。另一块无线网卡B会根据控制信道上所广播的信道信息而设置工作频段。普通服务数据只能在服务信道即无线网B上传输。

由于控制信道网卡A一直工作在固定的专用频率上，且这个频率会被限制保护，因此保证紧急重要数据的传输。对于普通服务数据，紧急重要性较低，使用上比较自由，可通过控制信道协调另一块网卡B选择空闲的服务信道进行传输。

上述无线双无线网卡系统还有一个好处：当控制信道网卡由于一些原因而不能工作时，服务信道网卡将取代控制信道网卡工作在控制信道上传输紧急重要数据传输。

如图三3所示，会有一个系统充当中心节点RSU，其余系统均为客户端。系统启动时，中心节点AP的控制信道网卡会定期的发送信道协调信息，以及根据需要发送紧急重要数据。客户端的控制信道网卡监听控制信道上的信息，加入网络。



图三3 系统图

传输紧急重要数据时，中心节点AP和客户端直接在控制信道上传输紧急重要数据。

传输大量普通服务数据时：

（1）中心节点发送

如果已经存在服务信道通讯，则直接在服务信道上请求发送。如果只建立有控制信道通讯，中心节点则直接在控制信道上发布信道协调信息，声明将要使用的服务信道频率。然后系统之间在服务信道上建立通讯进行大量数据传输。

（2）客户端发送

如果已经存在服务信道通讯，则直接在服务信道上请求发送。如果只建立控制信道通讯，则在控制信道上向中心节点发送请求。中心节点AP收到请求后，将根据需要广播服务信道频道值，最终客户端的服务信道网卡将在服务信道上建立通讯进行传输。当传输结束时，会通过控制信道网卡控制信道或服务信道通知AP普通服务数据传输已结束。

控制信道上，大部分数据是中心节点AP下行发送给各个客户端，各个客户端较少在控制信道上发送数据给AP，除了少量必须回应数据或请求数据。目的是尽量减少CSMA冲突几率。

普通服务数据不局限于AP和终端之间传输，它们也可以在终端之间进行传输。

如图三3所示，系统1、系统2在服务信道1上和AP进行普通服务数据传输，而系统3和系统4在另一个服务信道2之间进行数据传输。系统3和系统4服务信道使用完毕时，由于没有跟中心节点建立服务信道通讯，因此只能在控制信道上向中心节点通知。系统1和系统2除了可以在控制信道上向中心节点上通知和请求服务信道服务，也可以在当前服务信道上进行请求和传输。



图 三4 控制信道广播服务信道参数值

对图三3 来说，每个系统均使用了双无线网卡，均使用了两个信道。对整个图来说，使用了一个控制信道和两个服务信道。

* 1. 相关软件代码实现
     1. libpcap移植

libpcap[[[17]](#endnote-19)]是unix/linux平台下的网络数据包捕获函数包，大多数网络监控软件都以它为基础。Libpcap可以在绝大多数类unix平台下工作。

libpcap主要由两部份组成：网络分接头（Network Tap）和数据过滤器（Packet Filter）。网络分接头从网络设备驱动程序中收集数据拷贝，过滤器决定是否接收该数据包。Libpcap利用BSD Packet Filter（BPF）算法对网卡接收到的链路层数据包进行过滤。BPF算法的基本思想是在有BPF监听的网络中，网卡驱动将接收到的数据包复制一份交给BPF过滤器，过滤器根据用户定义的规则决定是否接收此数据包以及需要拷贝该数据包的那些内容，然后将过滤后的数据给与过滤器相关联的上层应用程序。

libpcap的包捕获机制就是在数据链路层加一个旁路处理。当一个数据包到达网络接口时，libpcap首先利用已经创建的Socket从链路层驱动程序中获得该数据包的拷贝，再通过Tap函数将数据包发给BPF过滤器。BPF过滤器根据用户已经定义好的过滤规则对数据包进行逐一匹配，匹配成功则放入内核缓冲区，并传递给用户缓冲区，匹配失败则直接丢弃。如果没有设置过滤规则，所有数据包都将放入内核缓冲区，并传递给用户层缓冲区。

libpcap的抓包框架是：

pcap\_lookupdev()函数用于查找网络设备，返回可被pcap\_open\_live()函数调用的网络设备名指针。

pcap\_open\_live()函数用于打开网络设备，并且返回用于捕获网络数据包的数据包捕获描述字。对于此网络设备的操作都要基于此网络设备描述字。

pcap\_lookupnet()函数获得指定网络设备的网络号和掩码。

pcap\_compile()函数用于将用户制定的过滤策略编译到过滤程序中。

pcap\_setfilter()函数用于设置过滤器。

pcap\_loop()函数pcap\_dispatch()函数用于捕获数据包，捕获后还可以进行处理，此外pcap\_next()和pcap\_next\_ex()两个函数也可以用来捕获数据包。

pcap\_close()函数用于关闭网络设备，释放资源。

本次实现libpcap移植到tiny6410开发板的过程：

（1）从http://www.tcpdump.org 网站上下载libpcap源码包，本此移植过程libpcap版本号为1.0.0。

（2）解压，然后进入到目录中，修改confiugre文件，把下面两段注释掉

#if test -z "$with\_pcap" && test "$cross\_compiling" = yes; then

# { { echo "$as\_me:$LINENO: error: pcap type not determined when cross-compiling; use --with-pcap=..." >&5

#echo "$as\_me: error: pcap type not determined when cross-compiling; use --with-pcap=..." >&2;}

# { (exit 1); exit 1; }; }

#fi

.......

# if test $ac\_cv\_linux\_vers = unknown ; then

# { { echo "$as\_me:$LINENO: error: cannot determine linux version when cross-compiling" >&5

#echo "$as\_me: error: cannot determine linux version when cross-compiling" >&2;}

# { (exit 1); exit 1; }; }

# fi

$ tar zxvf libpcap-0.9.5.tar.gz

$ cd libpcap-0.9.5/

（3）配置和编译

$ CC=arm-linux-gcc ac\_cv\_linux\_vers=2 ./configure --host=arm-linux --with-pcap=linux

$ make

修改Makefile中内容，/home/wangxing/armlinux/4.5.1/ 是电脑上交叉编译工具链目录。

prefix = /home/wangxing/armlinux/4.5.1/

（4）安装相关头文件和函数库libpcap.a放到arm-none-linux-gnueabi目录下相关文件夹里边去。之后就可以使用arm-linux-gcc来编译相关的libpcap应用的软件。

$ make install

（5）编译过程还产生了libpcap.so.1.2.1文件，这个文件通过串口下载到开发板中。

tiny6410 $:rz

SecureCRT: send libpcap.so.1.2.1 libpcap.a tcpdump

tiny6410:

$ mv ibpcap.so.1.2.1 /lib/ibpcap.so

* + 1. libpcap抓包收发示例

这里展示一个简单的无线网卡网卡收发例子。一个无线网卡会发送一个以下格式的内容：

“Packetspammer %02d roadcast packet#%05d -- :-D --%s ----",Rate/2, Ordinal, Hostname

最终效果如图所示，其中图三5是指x86架构的PC平台上无线网卡收发的，图三 6是tiny6410开发板接收到的。



图 三5 x86发送和接收反馈图



图 三6 Tiny6410开发板上接收到的数据

具体代码使用libpcap的抓包框架，即可实现。以下是其中所用到的数据结构体，从这些可看出，这种方法可定制任何的数据头部，然后让无线网卡直接发送出去。

static const u8 u8aRadiotapHeader[] = {

0x00, 0x00, // <-- radiotap version

0x19, 0x00, // <- radiotap header length

0x6f, 0x08, 0x00, 0x00, // <-- bitmap

0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, // <-- timestamp

0x00, // <-- flags (Offset +0x10)

0x6c, // <-- rate (0ffset +0x11)

0x71, 0x09, 0xc0, 0x00, // <-- channel

0xde, // <-- antsignal

0x00, // <-- antnoise

0x01, // <-- antenna

};

static const u8 u8aIeeeHeader[] = {

0x08, 0x01, 0x00, 0x00,

0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,

0x13, 0x22, 0x33, 0x44, 0x55, 0x66,

0x13, 0x22, 0x33, 0x44, 0x55, 0x66,

0x10, 0x86,

};

typedef struct {

int m\_nChannel;

int m\_nChannelFlags;

int m\_nRate;

int m\_nAntenna;

int m\_nRadiotapFlags;

} \_\_attribute\_\_((packed)) PENUMBRA\_RADIOTAP\_DATA;

* + 1. 双网卡平台系统

在这个系统中，无论是控制信道还是服务信道上的数据，在底层都是使用了如图三7 所示的数据帧格式。长度是一个16bit的字段，用于指示后续的类型ID的长度。不同类型ID其后续的有效数据内容不同。对于控制信道上的重要紧急数据，类型ID后续直接跟其内容。而对于服务信道上的应用数据，有效数据最开始的多少字节按照类型ID约定的字段是有效内容的长度。总之，这种数据帧方式简单，可扩展性强。这种底层的格式暂时只区分控制信道广播、控制信道内容单播等，对于上层的不同内容，比如安全信息中的交通事故、道路状况、车辆故障等，在有效数据中再次封装，由应用层封装和解封装。

数据链路头

长度

类型

ID

有效数据

数据链路尾

图 三7 数据帧格式

首先，网卡抽象层中用pthread\_create()[[[18]](#endnote-20)]启动两个线程，分别对两个无线网卡进行监控，使用libpcap专门负责底层的收发包，使用iwconfig来设置无线网卡的工作频道。这两个线程只接受上层封装好的数据，然后选择合适频道，封装好长度和类型ID，从底层无线网卡发送出去。接收时，根据长度和类型ID区分出控制信道上的CCH数据和服务信道的SCH数据。在这里，CCH数据的ID设为0x0F，SCH数据的ID设为0xF0。



图三8 平台抽象图

RSU上有一个线程，专门用来广播紧急事件，同时接收其他OBU上传的紧急事件。考虑到OBU会离开一个区域，所以OBU上传紧急事件之后，可选择由RSU代为广播紧急事件。在广播过程中，所有的RSU和OBU均会记住有哪些终端节点存在。



图 三9 发送安全广播数据

图中数据封装很直接，安全信息类中，用一个字节来表示，其中用0x01表示交通事故，0x02表示道路状况，0x04表示车辆故障，剩下的其他值作为扩展用。安全等级也是用一个字节表示，0x01表示特大，0x02表示严重，0x04表示重要，0x08表示普通，剩下其他值作为扩展用。是否重复、重复间隔时间和重复次数只是作为应用层序使用，不进行封装。接着是最后的消息摘要，首先是表示消息摘要的长度，如“Car Collision!”不含双引号的长度为14，即0x0e，接着是消息摘要的内容。对于上图的情况，发送到CCH上时的有效数据内容是

0x01 0x04 0x0e 0x43 0x61 0x72 0x20 0x43 0x6f 0x6c 0x6c 0x69 0x73 0x69 0x6f 0x6e 0x21



图 三10 接收到紧急广播数据

如图三10 所示，OBU接收到紧急广播数据时，会在屏幕上弹出一个窗口，同时用QSound bells("sound/alert.wav")播放提醒铃声。



图 三11发送服务数据

发送服务数据时，要指定发送内容，在这里暂时以文本文件和图片文件为例，指定发送对象，指定发送的服务信道。发送时，终端节点会现在控制信道上广播要发送的对象和发送的服务信道，待对方做好准备时，再在服务信道上发送服务内容。对方接收到后，将接收到的文本或者图片显示出来。



图 三12显示收到的服务数据

* + 1. 性能测试

在这种系统中，数据发送和接收受底层硬件、内核等元素影响，无法直接分析其速率性能。这里主要针对libpcap和TCP/UDP这两种做对比，从而得到性能上的感知。

在这次测试中，无论libpcap还是TCP/UDP，每次发包都选择相同的大小，比如1514B。单次测量发包的时间，受其他因素影响太大，所以改成测试多次累积时间。

首先测试读10M文件的时间，中间不加发送接收等其他调用。

gettimeofday(&start, NULL);

while( (ReadSize = read (fd, Buf, BUFSIZE)) >0)

{

//ret = pcap\_sendpacket(device,(const u\_char \*) Buf, ReadSize);

//if( ret != 0)

//{

// printf("sendpacket error %d %s\n", ret, pcap\_geterr(device));

// break;

//}

}

gettimeofday( &end, NULL);

printf("%ld\n",1000000 \* ( end.tv\_sec - start.tv\_sec) + end.tv\_usec - start.tv\_usec);

上述代码中已经把libpcap发送的代码注释掉了。

接着测试读10M文件并通过libpcap发送出去的时间。

gettimeofday(&start, NULL);

while( (ReadSize = read (fd, Buf, BUFSIZE)) >0)

{

ret = pcap\_sendpacket(device,(const u\_char \*) Buf, ReadSize);

if( ret != 0)

{

printf("sendpacket error %d %s\n", ret, pcap\_geterr(device));

break;

}

}

gettimeofday( &end, NULL);

printf("%ld\n",1000000 \* ( end.tv\_sec - start.tv\_sec) + end.tv\_usec - start.tv\_usec);

同样测试在TCP和UDP下的情况下的数据。对于这些测试，单次的数据均具有较大的波动性，所以实际操作中用shell脚本控制其自动运行多次，比如65536次，然后做平均值处理。接着测试20M等文件的时间。

#!/bin/bash

exec 1> result30M

for((i= 1;i< 65536;i++))

do

sudo ./speedtest wlan0 file30M

done

上述代码中设置shell的标准标准输出到result30M文件，其中每一行的结果就是读取或者加上发送的全部时间，多次运行得到大量的结果，对这些结果通过awk ‘{total+=$1} END{print total/NR}’ result30M得到平均值。



图 三13 ibpcap发包时间



图 三14 TCP发包时间



图三15发包时间

从这三个图中可观察到，libpcap的发包时间远小于TCP/UDP的发包时间。首先是TCP发送中有ack确认机制，发送一个包需要对方回应，这在广播安全紧急事故的是否非常不必要。在广播安全紧急事故的时候，RSU只需要把信息广播出去。而且是车载环境下，汽车终端节点随时变化，维持TCP的连接性也是不必要的。UDP的发包时间远小于TCP，但由于UDP协议的特性，仍比libpcap发包时间长。libpcap的发包时间和读文件在一个数量级，而TCP/UDP的发包时间远长于读文件的时间。libpcap方式中，时间更多消耗在对信息的处理。

* + 1. 改进

本次使用的是libpcap开发包，实际上还有另外一个开发包libpcap-mmap[[[19]](#endnote-21)]。 libpcap-mmap是libpcap的一个改进版本，它们捕获数据包的结构相同。不同的地方主要有以下两点：

1. libpcap使用固定大小的存储缓冲器和保持缓冲器来完成数据包从内核到用户层的传递，而libpcap-mmap设计了一个大小可以配置的循环缓冲器，允许用户程序和内核程序同时对该循环缓冲器的不同数据区域进行直接的读取。

2. 在libpcap中，当网卡接收到一个数据包之后，网卡驱动程序通过DMA方式调用系统函数netif\_rx()将数据包从网卡拷贝到核心态内存，应用程序想访问位于核心态内存的数据时就必须将数据包从核心态内存中拷贝到用户态内存中，这样就占用了很多系统资源，降低数据包捕获的性能以及对数据包的处理能力。而libpcap-mmap采用MMAP技术，建立核心态内存和用户态内存的映射，将系统分配给网卡设备文件的核心态内存映射到一块用户态内存，这样应用程序可以通过调用系统函数recvfrom()函数把数据包从网卡上直接传送到用户态内存中，减少了一次数据拷贝，降低了系统资源的消耗，提高数据包捕获效率。

另外一个，可以把这些应用层模块写成内核模块，在内核中做好各种类型判断、有效性判断，运行起来将更直接。

* 1. IEEE802.11p驱动移植概述

IEEE802.11p中有一种WAVE模式，就是WAVE设备可以不经过网络认证直接加入网络。由于具体的物理特性与具体实际网卡有关，所以这里不对具体相关展开，在这里主要阐述实现这种特性通用过程。

首先在Linux内核中新增一个变量dot11OCBEnabled，为IEEE802.11p的MAC层新定义属性，用来按指示设备发送数据帧时是否需要加入一个网络，然后用EXPORT\_SYMBOL\_GPL(dot11OCBEnabled); 这样子，内核中其他的模块比如驱动程序就可以访问这个变量。

1) 未加入网络的设备也可以发起通信

dot11OCBEnabled其默认值为假。设备正常运行时，如果此属性为假，则此设备需要加入到网络中才可以发送数据。

OBU发送probe request ->

<- RSU发送probe response

OBU发送身份验证Authentication request->

<- RSU发送Authentication Confirm

OBU发送关联Association request ->

<- RSU发送Association Confirm，注册一个用户

dot11OCBEnabled如果为真则此设备没有加入任何网络，但此时即可进行直接通信，通信功能及要求如下：

(1) 无需同步、认证、关联、指定为协议IEEE 802.11标准中11.1及11.3节所定义的其中一种帧类型，以及无需数据加密，此时设备可以即发送管理帧，包括IEEE 8011.11标准中7.4节定义的行动帧，及（如果设备具有一个TSF定时器）定时广播帧。

(2) 设备可以发送，除PS-Poll（省电-轮询）， CF-End（无竞争周期结束）， 和CF-End + CF-Ack（无竞争周期结束+无竞争周期确认）之外的控制帧。

(3) 设备可以发送 Null，Qos Data和Qos Null的数据帧。

(4) 设备在发送帧时设置BSSID域为通配符BSSID值（全1）。

(5) 另外此时通信时，MAC帧结构中ToDs域及FromDs域都要设为0。

|  |  |  |
| --- | --- | --- |
| 控制帧 | 管理帧 | 数据帧 |
| 除PS-Poll CF-End CF-End + CF-Ack之外 | 行动帧Action frame  （如果设备具有一个TSF定时器）定时广播帧 | Null，Qos Data和Qos Null |

表三1dot11OCBEnabled为真时设备可发送类类型

2) 具有相同的BSSID的设备，即在同一个网络中的设备，无需任何其它身份确认和连接即可直接进行通信

同一个网络（WBSS）中的所有终端设备具有相同唯一BSSID标识。当设备已经加入到网络中，即存储此网络的BSSID。在设备间进行通信时只要具有相同BSSID，则MAC层无需任何其它身份确认及连接即可直接通信，其它鉴权过程由上层定义。

* 1. 本章小结

本章主要介绍双无线网卡系统的设计方案，包括硬件及软件。本设计主要思路是硬件上利用现有的IEEE802.11网卡，软件上使用libpcap开发包提供收发链路层次的数据包的能力。在这个基础上，提出了网卡抽象层和事务抽象层的模型，定义了所使用的数据帧的格式。在这个平台上可以对车联网业务进行开发和验证。同时测试对比了libpcap和TCP/UDP方式的速度差异，验证了libpcap方式比TCP/UDP更适合紧急时间广播的优越性。接着对IEEE802.11p驱动提出了主要修改内容，以后在IEEE802.11p无线网卡成熟出现时，按照这个修改流程即实现IEEE802.11p驱动。本设计方案目前只是着眼于实现网络通信功能，暂时没有考虑性能、用户界面、产品化、安全等问题。

1. 基于TDMA的媒体接入控制设计

无线网络中包含许多的无线终端节点，空间中传输的无线电波则是无线网络的媒介。由于无线电波的特殊性，在某个频率上同一时刻只能有一个无线终端发射无线电信号，其他的终端接收信号。各个无线终端在竞争无线媒介发射无线电信号时，需要遵循一定的媒体访问控制方法MAC。无线网络MAC协议的设计主要有两大类：基于竞争的随机媒体访问和基于调度的媒体访问。

基于竞争的随机访问MAC协议采用按需使用信道的方式，IEEE802.11系列都采用了这种方式。当节点需要发送数据使用无线媒介发射无线电波时，通过竞争方式使用无线信道，如果在发射无线电波时检测到有其他的无线终端在使用无线媒介，产生了碰撞，就按照某种策略退避等待，然后重新发送数据，直到：（一）数据发送成功；或者（二）放弃发送数据。这种方式的主要好处是当网络中的节点数目较少和数据流量较小时，能够提高无线信道的利用率，同时也无需节点时钟同步，算法设计简单。缺点是当网络中源节点较多且网络中的数据流量较大时，多个节点频繁地竞争无线媒介信道，发生碰撞的概率也较大，容易导致节点发包失败的越大。因此提出了TDMA的媒体访问信道接入方法。

* 1. TDMA概述

基于调度的媒体访问控制协议主要是采用时分复用TDMA（Time Division Multiple Access）的方法，按照事先约定的策略周期性地给各个节点分配一定的时槽（slot），用于数据发送或接收。终端节点的无线电天线在其他空闲非自己时槽时保持接收或者睡眠状态。TDMA机制少了竞争机制的碰撞重传问题。当网络中数据流量较大，无线媒介竞争较为严重的情况下，TDMA机制是一种较好的无线媒介控制机制。缺点是TDMA机制需要无线终端节点之间的严格时隙同步，时隙分配算法有一定的复杂度。时隙同步一般可通过GPS来解决，或者使用广播帧的方式。TDMA一般主要是研究时隙分配算法。

信道接入协议有很多种分类。同步信道协议协议是指所有终端节点在时间上是同步的；异步信道接入协议是基于竞争模式的。收方驱动协议是收方通知发方自己已经准备好接收数据；发方驱动是发方通知收方自己又数据需要发送。单信道信道接入协议是所有的信号在同一个信道上传输；双信道信道接入协议有两个共享信道，分别为控制信道和数据信道；多信道信道接入协议是有多个信道，相邻可使用不同信道同时进行通信。固定分配类协议按照网络中最大节点数量做出传输时间安排，为每个用户分配确定的信道；动态分配类协议使用本地参数来分配时隙，根据时隙分配表的产生过程与拓扑结构的依赖关系可分为拓扑非透明算法、拓扑透明算法和拓扑半透明算法。



图四1 不同时隙安排图

如图四1所示，车A的前面发生了紧急事件，这个事件需要通过信息传播给后续车辆B、C、D。按照a）的时隙安排，Frame1中，C和B发送的数据不包含紧急事件，A发送后B才收到紧急事件，而C、D不在A的有效传输范围内而无法感知到紧急事件。接着在Frame2中，C发送的数据中仍没有紧急事件，而B发送的数据中有，相当于在Frame2中讲紧急事件传输给C了。最后再Frame3中，C发送了数据才将紧急事件通知给了D。

而在b)中情况很不一样。A发送了数据，告知B；轮到B发送的时隙时，B已经知道了事件，所以可以发送给C；C发送时也知道了事件，所以可以直接告知D发生了紧急事件。整个过程仅在Frame1即可完成。

从这个例子可看出，不同的时隙分配算法会有不同的效果[[[20]](#endnote-22)]。由于车载网中，车载终端做为无线终端，数目较多，因此研究TDMA方式是很有必要的。在车载环境中，常见的情境有自组织网络，是由多个车载无线终端组成的无中心、多跳的临时自治系统。这种网络无需基础设施，可快速组网。一般情况下每个节点只能感知其一跳范围内的邻居节点的信息，存在明显的节点隐藏和暴露节点问题。

如果有两个终端节点同时使用同一时隙，那么会导致冲突发生。时隙访问如何尽量避免冲突是一个重要的问题。于是本次论文提出了位置信息辅助的多路访问方案。

* 1. 位置信息辅助的多路访问方案



图四2位置信息辅助的多路访问方案情景图

在这里假设时间带宽被分为M个独立正交的时隙，每个时隙只能分配给一个车载终端节点使用。多个车载终端节点占用同一时隙则视为冲突，将没有一个终端节点能成功收发数据。所有车载终端节点过GPS或其他途径来获得地理位置。一般来说，每个车载终端节点的位置是独立的，和其他的车载终端节点位置互不干扰互不重叠的。

由于车辆地理信息的唯一性，因此可以根据汽车终端节点的地理位置来分配时隙。一个位置-信道的映射算法用来帮助汽车终端节点选择时隙。



图四3 位置区域

如图四3 所示，将汽车终端节点地理位置抽象成坐标，分为两维，一维是汽车的行走方向，即x轴方向，另一维是垂直车行走方向，即y轴方向。最大值K则由道路匝数决定。普遍来说，上述坐标的形成过程分为三部分：

1）一个主汽车终端节点要发送数据，在它的无线通信范围内广播。

2）其他汽车终端节点收到广播的数据时，会回应它们的地理位置信息给主汽车终端节点。

3）主汽车终端节点根据收到的地理位置信息来编号，然后再把编号信息发送出去。

在这里，其他汽车终端节点在回应地理位置信息时，它们是什么时候发送，这个问题在这里简单描述一下。一般时隙划分后，前面大部分是分配给各个终端节点的，而后面会有小部分是公共时隙，公共时隙是给新增或新加入节点随机竞争访问的。一开始网络在形成时，数目还是比较小，这种情况下，几个节点在公共时隙上随机竞争，一般很快就能被分配到前面的专有时隙。随着网络的节点增多，这些节点也是慢慢地通过公共时隙获得发言权，然后被分配到专有时隙。而且，在车载网络环境下，汽车终端节点拓扑的变化，远远小于终端节点的通讯速率，可能几分钟才有一辆车载终端节点加进来，而一个完整时隙轮完一遍，可能时间也就在毫秒级别，相比之下，车载终端节点拓扑的变化，可视为缓慢而稳定，在某种程度上，可暂时忽略终端节点拓扑的变化。

一般来说，时隙数目M是小于K\*N的，所以多个终端节点可能会访问同一时隙，导致冲突发生。位置信息辅助的多路访问方案使用了地理位置信息来帮助终端节点有效地分配时隙。在这个方案中，一个终端节点会以不同的概率来访问时隙，而非随机访问。有以下规则：

定义了一个专用的函数来将汽车终端地理信息L(i,j)映射到一个随机变量Y，随机变量Y表示被划分的正交的M个时隙的序列号。函数可以表示为：



其中，表示由k-th地理位置所映射的随机变量序列号，表示k-th位置映射用到的函数，e是随机变量里的一个未知数。L(i,j)表示车载终端节点的位置。K表示道路匝道数目，N表示区域里边位置的数目。

每个随机变量Y满足一定的概率函数：





其中，M表示被划分的时隙数目。表示汽车终端节点访问m-th时隙的概率。

3）为了满足分布式时隙访问的要求，位置信息和概率分布必须满足以下条件：

A）对于任意区域，映射是唯一。换句话说，只要区域已形成，位置就已经确定了，映射也就同时确定了。

B）映射只依赖于概率分布和汽车终端节点的相对位置，与通过GPS获得的绝对位置无关。

4)）如果不同位置的车载终端节点访问相同的时隙，那么冲突发生，在这种情况下认为没有车载终端节点能成功收发数据。这种情况可以用表达式表达：



其中，和是随机变量、对应的值。

* 1. 冲突概率和吞吐量
     1. 冲突概率

不失一般性，这里假设情景发生在只有一个车道，即K=1。区域长度L则取决于能保证该区域中能正常与其他车辆通讯的有效距离。我们假设区域中每个位置都有一个唯一的索引值，比如1,2,…,N。同时，任意车辆出现在每个位置的概率是相等，用表示。车载终端节点在位置N处访问第M-th的概率用。例如图四4 、、。



图四4时隙划分和车载终端节点访问

基于以上假设，这里总共会出现N\*M个未知参数，(𝑖 = 1, 2, . . .,𝑁; 𝑗 =

1, 2, . . .,𝑀) ，我们使用了遗传算法来优化这N\*M个参数，优化目标是降低这个区域中平均冲突可能性。这个可能性的优化值会被当做车载终端节点访问时隙的基准。换句话说，当车载终端节点需要占用时隙来收发数据时，它必须根据来决定占用哪个时隙。这里我们和一般随机访问方案作对比。为了两种方案做对比，必须计算一些性能指标来评估对比。我们可以计算它们对应的平均冲突概率和吞吐量。两种方案的平均冲突概率和吞吐量具体的计算方式如下：

位置信息辅助的多路访问方案的平均冲突概率：

按照上述提到的，车载终端节点出现在任何一个地理位置的概率是相等的，在这个方案中，车载终端节点在每个位置中访问不同的时隙的概率是不同的。车载终端节点在访问时隙时，有可能会遇到其他车载终端节点访问同一时隙而引发的冲突。首先，我们计算车载终端节点在每个位置中访问时隙的概率分布，然后把这些加起来得到总的概率分布，最后除以N，就得到了平均的概率分布。为了得到平均概率分布的表达式，我们首先计算一个车载终端节点在一个地理位置的概率。在这个位置中，一个车载着终端节点可能随机访问一个时隙。如果该车载终端节点访问的时隙也被其他车载终端节点访问，那么冲突发生。车载终端节点在一个地理位置1的冲突概率模型可表示为：

 式 1



式 2

其中，表示车载终端节点在一定位置1访问时隙时的冲突概率，表示车载终端节点在第一个位置访问时隙时非冲突的概率，N表示讨论区域中位置的数目，M表示时隙被划分的数目，表示一个车载终端节点出现在某个位置的概率。

可表示为

 式 3

根据上述的表述，车载终端节点在所有位置的总的冲突概率表达式可表示为：

 式 4

 式 5

 式 6

其中，表示讨论区域中所有车辆访问时隙时总的冲突概率，表示讨论区域中所有车辆访问时隙时非冲突的概率，代表讨论区域中所有车辆访问时隙时的平均冲突概率。

随机访问方案的平均冲突概率：

随机访问方案是一种很简单的车载终端节点访问时隙的方案。在这个方案中，所有的车载终端节点访问时隙的概率都是相等的，即1/M。这种方案下车载终端节点访问时隙的平均概率等于车载终端节点在任何一个位置访问时隙的冲突概率。因此，平均冲突概率模型为：

 式 7

* + 1. 吞吐量

为了进一步比较这两种方案，我们提出了吞吐量这个性能指标。这里有些前提假设，首先是时隙条件状况是稳定和统一的。第二，吞吐量是通过车载终端成功占用某一时隙平均传送的消息来估计的。基于以上假设，我们这里可以得到吞吐量的表达式：

 式 8

其中，表示时隙的平均吞吐量，C表示时隙带宽容量，M表示时隙的数目，表示一个车载终端节点访问时隙时的冲突概率。

* 1. 建模求解

根据式 5 和式 6，我们可以看到平均冲突概率是依赖于位置-时隙的映射。我们应该优化位置-时隙映射的参数来获得最小的平均冲突概率。优化问题的目标函数和约束函数可以改写成：

目标函数

 式 9

（2）约束条件

 式 10

位置信息辅助的多路访问方案中的总冲突概率表达式中含有多个未知变量、高阶、非线性的特征。这些特征使得优化变得更加困难。同时，常见的优化解决方法无法达到这个模型的优化目标。遗传算法优化是一个以自然选择进化论的自适应、启发式的算法，在一定搜索范围内利用智能随机搜索来解决优化问题，能搜索到一个近乎全局优化的解决算法。遗传算法的基本思想是基于达尔文的适者生存的原则。按照这个原则，遗传算法分为以下步骤[[[21]](#endnote-23)][[[22]](#endnote-24)]：

a)初始化：设置进化代数计数器t=0，设置最大进化代数T，随机生成M个个体作为初始群体P(0)。

b)个体评价：计算群体P(t)中各个个体的适应度。

c)选择运算:将选择算子作用于群体。选择的目的是把优化的个体直接遗传到下一代或通过配对交叉产生新的个体再遗传到下一代。选择操作是建立在群体中个体的适应度评估基础上的。

d)交叉运算：将交叉算子作用于群体。遗传算法中起核心作用的就是交叉算子。

e)变异运算：将变异算子作用于群体。即是对群体中的个体串的某些基因座上的基因值作变动。

群体P(t)经过选择、交叉、变异运算之后得到下一代群体P(t+1)。

f)终止条件判断:若t=T，则以进化过程中所得到的具有最大适应度个体作为最优解输出，终止计算。

通过上述的遗传算法的迭代，将获得一组个体，使得总的冲突概率获得最小值。在上述过程中，将获得总冲突概率函数的各个参数的值，这些值使得函数达到近乎全局优化。

在这次matlab仿真中，使用了雷英杰的genetic遗传算法库[[[23]](#endnote-25)]，定义了群体中个体数目（Number of individuals）为NIND =80，变量数目NVAR则由时隙数目和位置数目乘积决定，最大遗传代数（Maximum number of generations）为MAXGEN=4000，变量的二进制位数（Precision of variables）为PRECI =20，代沟（Generation gap）为GGAP=0.9。

仿真程序的主体框架为：

Chrom = crtbp(NIND, NVAR\*PRECI); %初始种群

gen=0; %代计数器

trace = zeros(2, MAXGEN);

X = bs2rv(Chrom, FieldD); %计算初始种群的十进制转换X(NIND,NVAR);

ObjV = objV\_calc(X, M, N, p); %计算初始种群的目标函数值

while gen < MAXGEN

FitnV = ranking(ObjV); %分配适应度值（Assign fitness values）

SelCh = select('sus', Chrom, FitnV, GGAP); %选择

SelCh = recombin('xovsp', SelCh, 0.7); %交叉

SelCh = mut(SelCh); %变异

X = bs2rv(SelCh, FieldD); %子代个体的十进制转换

ObjVSel = objV\_calc(X, M, N, p); %计算子代的目标函数值

[Chrom ObjV] = reins(Chrom,SelCh,1,1,ObjV,ObjVSel); %重插入子代的新种

gen = gen + 1; %代计数器增加

%输出最优解及其序号，并在目标函数图像中标出，Y为最优解，I为种群的序号

[Y,I]= min(ObjVSel);

trace(1, gen) = min(ObjV); %遗传算法性能跟踪

trace(2, gen) = sum(ObjV) / length(ObjV);

end

在这个遗传算法库中，创建种群的方法有ctrbase创建基向量、ctrbp创建任意离散随机种群和crtrp创建实值初始种群，在这里使用了crtbp函数创建了任意离散随机种群。

适应度计算中提供了ranking基于秩的适应度计算和scaling比率适应度计算，在这里使用了ranking基于秩的适应度计算函数。

选择函数中提供了reins一致随机和基于适应度的冲插入、rws轮盘选择、select高级选择例程和sus随机遍历采样，在这里使用了sus随机遍历采样。

交叉算子中提供了较多的选择，有recdis离散重组、recint中间重组、reclin线性重组等，在这里使用过了xovsp单点交叉算子。

变异算子有mut离散变异、mutate高级变异和mutbga实值变异，在这里使用过了mut离散变异。

适应度的计算过程为：

% 把随机生成的概率值单位化，使其满足：pn1+pn2+...+pnM = 1,n = 1...N;

for row = 1:NIND

for n = 1:N

for m = 1:M

sum = sum + X( row, (n-1)\*M + m );

end

for m = 1:M

Prob\_Mtrx( row, (n-1)\*M + m ) = X( row, (n-1)\*M + m ) / sum;

end

sum = 0;

end

end

% 计算个体的目标函数值

for i = 1:NIND

objV(i,1) = dest\_fun(Prob\_Mtrx(i,:), M, N, p);

end

* 1. 仿真结果

在这里，将比较两种不同方案的性能。在仿真中，泊松点过程用来描述车载终端节点的位置分布。车载拓扑变化的频率远远低于无线通信的速率。因此我们可以在一个快照中观察到分布。假设车载在这个区域中随机分布了。每个车载在任何时刻都有一个唯一的位置信息（即：不会发生位置信息重叠现象），车载出现在一个地理位置的概率是一个常数（即 ）。同时假设车载之间的连接是由车载之间的距离。为了改进无线时隙的利用率，时隙的划分数目M应该尽可能小。但是，越小的M值将带来更大的平均冲突。当吞吐量达到最大值，时隙数目M就是最优值。由于实际上车辆之间要保持一定的安全距离，所以设置了每辆车的长度为10米。同时，为了保证车与车之间能通讯，设置了区域长度为500米，这是IEEE802.11p的经典通讯距离。参考DSRC，设置了信道的通信容量C为20Mbps。所以，位置的数目N为50。值的变化，会很大地影响到平均概率。这里挑选了两个值。小的值代表了道路上比较少的车，大的值代表了车路上比较拥挤。在这仿真中，分别设置为0.3和0.7。在遗传优化算法的过程中，使用了格雷码编码方法来实现参数编码，一个参数使用了20bit来表示。还有，我们选择了总冲突概率函数最为最适应函数，基于排序方法来赋予最佳值。交叉和变异的概率分别设置为0.7和0.035。

首先，我们先设置车载终端节点出现在一个位置的概率为。我们观察时隙划分数目M的变化时，时隙的平均吞吐量和平均冲突概率的变化。从图5中，我们可以看到，在随机访问方案中，当M取15的时候，平均吞吐量达到最大值。而在位置信息辅助的多路访问方案中，平均吞吐量要高出16%。



图四5 吞吐量仿真图（N=50，）

从另一方面，从图四6 我们可以观察到位置信息辅助的多路访问方案中平均冲突概率总是低于随机访问方案。特别是，在吞吐量最大值（即M取17时），相比降低了13.7%，同时，位置信息辅助的多路访问方案的吞吐量比随机方案高出19%。



图四6平均冲突概率图（N=50，）

其次，我们修改了车载出现在一个位置的概率（即），然后观察随着时隙数目M的变化，平均吞吐量和平均冲突概率的变化。



图 四7 吞吐量仿真图（N=50，）

从图四7看到，随机访问方案中，最大的平均吞吐量出现在M取35的时候。在这个地方，位置信息辅助的多路访问方案的平均吞吐量高出了83%。两个方案的最大值均小于时的情况。



图 四8平均冲突概率图（N=50，）

同时，从图四8看到，位置信息辅助的多路访问方案中平均冲突概率仍然总是低于随机访问方案。两个方案的平均冲突概率下降均时的情况。

随着M的增长，平均冲突概率的降低就越明显。随着的增大，说明车况越来越拥堵，平均冲突概率增大，平均吞吐量下降。

* 1. 本章小结

在这里，提出了位置信息辅助的多路访问方案。在这个方案中，每个车载终端节点是根据位置-时隙映射表来选择时隙的，平均冲突概率分布降低了很多，平均吞吐量得到明显的提高。通过matlab遗传算法仿真结果表明，位置信息辅助的多路访问方案的平均吞吐量优于随机访问方案，随着时隙数目M的变化，其平均吞吐量和平均冲突概率也有不同的变化。

总结与展望

当前，世界各国研究机构和厂商都在开展有关车联网系统的研究。车联网系统是未来发展的一个趋势，是促进物联网发展的重要方面，它将改变人类的生活和生产。因此，研究车联网，对于智能交通、智能城市的发展等具有非常重要的意义。

IEEE802.11p和IEEE1609协议族做为车联网的主要通信协议栈，正在逐步完善。虽然要在实际中大规模应用和部署还有很长的路要走，更多的研究工作要完成，但是这也是一个发展的机会。

本文在学习和理解IEEE802.11p和IEEE1609协议族情况下，重点抓住多信道协调机制，在现有硬件条件尚不不完善的情况下，使用Tiny6410开发板和RT3070无线网卡，移植libpcap库，利用libpcap开发框架实现了一个双无线网卡通信平台系统，取得不错的使用效果。同时，深入IEEE802.11p协议和Linux内核编程，对实现IEEE802.11p驱动提出一个轮廓，为以后实现IEEE802.11p驱动提供了一个指导性的方向。最后文章还对TDMA做了研究，提出位置信息辅助的多路访问方案，给出平均冲突概率和吞吐量两个性能指标，使用了遗传算法求解优化结果，通过仿真验证了所提方案在两个性能指标方面的优越性。

当然，本文所做的工作有限，车联网是一个庞大的系统，需要投入更多的研究，系统还需更多完善：

（1）为了提高车载的通讯能力，需要选用专门针对车辆网的无线网卡，在硬件上实现相关的处理，然后把信道协调等功能实现在内核中，提高效率，抽象化接口，简化上层开发应用人员开发工作。

（2） 本文没有对路由方面做相关研究，而实际应用中，路由算法也是一个很关键的方面，一个好的路由算法，能更快地做出一个更优的路由选择，减少车载中的不必要的路由和转发。

（3）位置信息辅助的多路访问方案中，需要考虑一个更实际的车辆的随机分布模型，同时，无线电信号的衰减和衰弱模型也必须考虑进去。

总之，本论文研究了车联网应用的一部分，为整个车联网的研究和实现提供了一定的参考，未来可以此基础上继续完善和丰富，最终实现车联网，实现智能交通。

参考文献

[1] 百度百科.物联网. [EB/OL]. http://baike.baidu.com/view/1136308.htm

[2] 宋俊德. 浅谈物联网的现状和未来[J]. 移动通信. 2010,15

[3] 孙其博,刘杰,黎羴,范春晓,孙娟娟. 物联网:概念、架构与关键技术研究综述[J].北京邮电大学学报.2010,03

[4] 百度百科.智能交通系统[EB/OL]. http://baike.baidu.com/view/1488750.htm

[5] 王笑京,沈鸿飞,汪林. 中国智能交通系统发展战略研究[J]. 交通运输系统工程与信息. 2006,04

[6] 王建强, 吴辰文,李晓军. 车联网架构与关键技术研究[J].微计算机信息.2011,04

[7] Grafling, Sebastian, Petri Mahonen, Janne Riihijarvi. Performance evaluation of IEEE 1609 WAVE and IEEE 802.11 p for vehicular communications. Ubiquitous and Future Networks (ICUFN), 2010 Second International Conference on. IEEE, 2010.

[8] 常促宇,向勇,史美林.车载自组网的现状与发展[J]. 通信学报.2007,11

[9] 刘富强,项雪琴,邱冬.车载通信DSRC技术和通信机制研究.上海汽车.2007,08

[10] 迟铁军,高鹏.国外智能交通系统发展状况分析及对我国的启示.黑龙江交通科技.2009,02

[11] 美国ITS的发展历程与现状[EB/OL]. http://www.itschina.org/article.asp?articleid=2143.

[12] Tsugawa S. The Current Trends and Issues on ITS in Japan: Safety, Energy and Environment. 2011 IMWS-IRFPT[C]. 2011

[13] 谢鹏.基于WAVE协议栈的车联网通信终端研究及定位系统实现[D].华南理工大学.2013

[14] 曹颖荣,林小玲. IEEE 802.11p无线车载自组网络协议的性能分析与模拟[J]. 仪表技术, 2011,02

[15] 杜英田. 基于IEEE802.11p/1609协议的智能交通无线车载通信协议优化研究[D].北京邮电大学, 2011

[16] 百度百科.OSI模型[EB/OL]. http://baike.baidu.com/view/571842.htm

[17] 加斯特. 802.11无线网络权威指南(译) [M]. 东南大学出版社,2006

[18] 百度百科.IEEE802[EB/OL].http://baike.baidu.com/view/126269.htm

[19] 唐波. WAVE架构及相关协议设计与实现[D]. 华东理工大学,2009

[20] 东南大学.802.11无线网络[M]. 东南大学出版社,2006

[21] S. Eichler. Performance evaluation of the IEEE 802.11p WAVE communication standard[C]. Proc. IEEE Vehicular Technology Conf., Baltimore, MD, US, Oct. 2007: 2199-2203

[22] Karagiannis G.. Vehicular Networking: A Survey and Tutorial on Requirements,

[23] Architectures, Challenges, Standards and Solutions[J]. Communications Surveys & Tutorials, IEEE,Fourth Quarter 2011: 584-616

[24] IEEE 1609 - Family of Standards for Wireless Access in Vehicular Environments (WAVE) [EB/OL]. http://www.standards.its.dot.gov/fact\_sheet.asp?f=80

[25] IEEE Std 1609. 0-2012. IEEE Draft Guide for Wireless Access in Vehicular Environments (WAVE) - Architecture. IEEE P1609.0/D5[S]. IEEE, 2012

[26] IEEE Std 1609.1 -2006. IEEE Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE)—Resource Manager. IEEE Vehicular Technology Society (VTS) [S].October 2006

[27] IEEE Std 1609. 3-2010, IEEE Standard for Wireless Access in Vehicular Environments (WAVE)-Networking Services[S].IEEE, 2010.

[28] IEEE Std 1609. 4-2010，IEEE Standard for Wireless Access in Vehicular Environments (WAVE)-Multi-channel Operation[S].IEEE, 2011

[29] 邱团准. 基于SOA的车联网应用系统设计与实现[D]. 华南理工大学, 2013

[30] 百度百科.嵌入式系统[EB/OL]. http://baike.baidu.com/view/6115.htm

[31] 韦东山.嵌入式Linux应用开发[M]. 人民邮电出版社,2003

[32] 魏永明,耿岳,钟书毅. LINUX 设备驱动程序(译)[M]. 中国电力出版社,2006

[33] 百度百科.libpcap[EB/OL]. http://baike.baidu.com/view/1319961.htm

[34] Richard Steve.Advanced Propramming in the Unix Environment[M].人民邮件出版社2006

[35] Csdn博客.libpcap-mmap [EB/OL]..http://blog.csdn.net/aaa6695798/article/details/4451582

[36] Yu, Fan, and Subir Biswas. Self-configuring TDMA protocols for enhancing vehicle safety with DSRC based vehicle-to-vehicle communications. IEEE Journal. 2007,25

[37] 百度百科.遗传算法[EB/OL]. http://baike.baidu.com/view/45853.htm

[38] 遗传算法演示程序[EB/OL].http://userweb.eng.gla.ac.uk/yun.li/ga\_demo/ga\_demo.html

[39] 雷英杰,张善文,李续武,周创明MATLAB 遗传算法工具箱与应用[M].西安电子科技大学出版社,2005

攻读硕士学位期间取得的研究成果

一、已发表（包括已接受待发表）的论文，以及已投稿、或已成文打算投稿、或拟成文投稿的论文情况**（只填写与学位论文内容相关的部分）：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **序号** | **作者（全体作者，按顺序排列）** | **题 目** | **发表或投稿刊物名称、级别** | **发表的卷期、年月、页码** | **相当于学位论文的哪一部分（章、节）** | **被索引收录情况** |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

注：在“发表的卷期、年月、页码”栏：

1如果论文已发表，请填写发表的卷期、年月、页码；

2如果论文已被接受，填写将要发表的卷期、年月；

3以上都不是，请据实填写“已投稿”，“拟投稿”。

不够请另加页。

二、与学位内容相关的其它成果（包括专利、著作、获奖项目等）

致谢

白驹过隙，转瞬即逝，依稀记得刚进实验室时懵懵懂懂的我，在实验室老师们的指导下，不断地学习与成长。三年中，个人经历了各种起伏，如今的技术能力，也是集中在这三年中学成锻炼。如今三年的研究生生活即将结束，我衷心感谢所有帮助过和关心过我的人们。

首先感谢我的研究生导师吴宗泽教授，吴老师是我进入科研领域的领路人，他对科学研究的热情和孜孜不倦的研究精神大大感染了我，使得我对科研产生了兴趣。吴老师严谨认真的学术态度，矢志不渝的学术追求，创新求是，扎扎实实的学术精神，思维活跃的学术观点，营造了一种良好的学术氛围，使得置身其中的我被深深感染。在此，衷心地感谢吴老师，是吴老师您让我进入到这个充满温暖互相帮助团结的集体中。

其次，感谢实验室的余老师，他是我们项目组的指导老师，他认真负责的工作态度、专业的解决问题的能力，以及对学生的耐心指导，让我在做项目的过程中收获颇丰。余老师也不乏幽默，为人亲切，对我们学生就像学长般一样，让我感到亲切。余老师既是良师又是益友，他踏实肯干的工作态度和积极向上的生活态度，深深地鼓舞着我，让我在迷茫的时候能够找到方向。

感谢实验室其他的各位老师，他们为我的学习和发展提供了一个很好的平台。感谢实验室的众多师兄师姐们，他们带领我入门，为我提供许多指导与帮助。

感谢实验室的同学们（J神，蓉姐，容姐，D神，Y神，星爷，尖子，沈大，波导，涛哥，爽姐，文龙，芳芳等）在学习上，大家营造了良好的学习环境和学术氛围，在生活上，大家相互关心相互帮助，正是你们，才使我的三年研究生生活收获丰富、充满精彩。与你们同学三年，我亲身感受到大牛是怎么炼成的，你们也是一直以来我的学习榜样。感谢实验室的师弟师妹们，有你们的努力付出，才有我们的功成身退。

衷心感谢我的父母兄姐，他们一直都在背后默默地支持着我，他们从来尊重我的选择，有了他们的支持，我才能安心地完成所有的大学生涯，他们的关怀和支持将是我永远的精神支柱。

“路漫漫其修远兮，吾将上下而求索”，研究生阶段的结束只是新生活的开始，以后的日子里充满未知，但是我总记得有那么一些好老师、好同学，伴我度过了难以忘怀的三年，教会了我许许多多的东西，让我能够更加从容面对未来

1. [↑](#endnote-ref-3)
2. [↑](#endnote-ref-4)
3. [↑](#endnote-ref-5)
4. [↑](#endnote-ref-6)
5. [↑](#endnote-ref-7)
6. [↑](#endnote-ref-8)
7. , [↑](#endnote-ref-9)
8. [↑](#endnote-ref-10)
9. [↑](#endnote-ref-11)
10. [↑](#endnote-ref-12)
11. [↑](#endnote-ref-13)
12. [↑](#endnote-ref-14)
13. [↑](#endnote-ref-15)
14. [↑](#endnote-ref-16)
15. [↑](#endnote-ref-17)
16. [↑](#endnote-ref-18)
17. [↑](#endnote-ref-19)
18. [↑](#endnote-ref-20)
19. [↑](#endnote-ref-21)
20. [↑](#endnote-ref-22)
21. [↑](#endnote-ref-23)
22. [↑](#endnote-ref-24)
23. [↑](#endnote-ref-25)