资深网络技术工程师、全国网管技能水平考试认证专家、国内优秀IT图书作者王达老师最新力作，51CTO技术社区鼎力推荐！

结合最新技术，全面、系统、深入阐述计算机网络的体系结构、功能实现原理和通信协议实现原理

包含近600幅图表、形象的比喻和丰富的案例，使得本书通俗易懂，极大程度地降低了读者的学习成本

提供教学用PPT

华章原创精品

深入理解计算机网络

Understanding Computer Networks

王达 著

机械工业出版社
China Machine Press

深入理解计算机网络

王　达　著

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

深入理解计算机网络 / 王达著 . —北京：机械工业出版社，2013.1

ISBN 978-7-111-41188-8

I. 深…　II. 王…　III. 计算机网络　IV. TP393

中国版本图书馆 CIP 数据核字（2013）第 009187 号

版权所有・侵权必究

封底无防伪标均为盗版

本书法律顾问　北京市展达律师事务所

本书是计算机网络领域的扛鼎之作，由有 20 余年从业经验的优秀网络技术工程师兼全国网管技能水平考试认证专家王达老师撰写，51CTO 技术社区鼎力推荐，权威性毋庸置疑。内容方面，本书结合最新计算机网络技术，全面、系统、深入地阐述了计算机网络的体系结构、工作原理，以及各种通信协议实现原理，能满足读者系统和深入地学习和研究计算机网络技术的需求。阅读体验上，近 600 幅图表、形象的比喻和丰富的案例使得本书通俗易懂，能极大地降低学习难度。除此之外，为了便于教师教学，本书还提供了精心制作的教学 PPT。

全书共 11 章：第 1 章详细介绍了数制与编码的相关知识；第 2 章宏观地讲解了计算机网络的组成、应用、分类，以及计算机网络的拓扑结构；第 3 章深入地讲解了典型的计算机网络体系结构、计算机网络体系结构的通信原理和通信协议，以及网络体系结构设计时的考虑；第 4 ～ 7 章和第 10 ～ 11 章分别系统且深入地讲解了物理层、数据链路层、介质访问控制子层、网络层、传输层和应用层的作用、技术细节和实现原理；第 8 章深入地探讨了 IP 地址和子网，不仅讲解了 IPv4 相关技术，也对最新的 IPv6 相关技术做了深入的探讨；第 9 章系统介绍了 RIP、OSPF、IS-IS、BGP 等各种路由协议及其实现原理。

本书既适合作为想全面深入了解计算机网络技术的网络工程师们深入学习的资料和工作时的参考资料，又适合作为各高等院校的老师和学生们系统学习计算机网络技术的教材。

机械工业出版社（北京市西城区百万庄大街 22 号　　邮政编码　100037）
责任编辑：孙海亮
印刷
2013 年 1 月第 1 版第 1 次印刷
186mm×240mm • 40.75 印张
标准书号：ISBN 978-7-111-41188-8
定　　价：89.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
客服热线：(010) 88378991　88361066　　投稿热线：(010) 88379604
购书热线：(010) 68326294　88379649　68995259　　读者信箱：hzjsj@hzbook.com

前　言

本书是笔者从业二十余年、从事图书创作十余年的工作经验和技术积累的结晶，是对十余年来一直默默支持我的全国百万读者的真诚回馈。同时，本书也是笔者这十多年来付诸心血最多（整整一年专职创作时间）、寄予希望最大的一部重头之作，期待能为国家的计算机网络专业教育尽一份绵薄之力。

为什么写这本书

其实很久以前就有了写这本书的动机了，但由于我深知写作这本书的难度很大，再加上自己还在写作其他图书，写作任务一直非常繁重，所以就耽误了。不过，或许今天写这本书正是时候，一则笔者又多经过了几年的技术学习和工作经验的积累，书稿的质量可能比以前更高；二则目前计算机网络专业越来越边缘化了，已成为了所有 IT 人员的必修课，所以现在对要求有一本高质量、通俗易懂的专注于计算机网络原理和基础知识的教材的呼声比以前更高了。综合起来就是以下三点：

1. 学子的呼唤：“零基础”不应只是一句宣传口号

计算机网络原理和基础知识类的课程一直是广大计算机网络专业的读者最头痛的一门课程。因为这类课程不仅相对枯燥乏味，而且教材中的技术原理解释普遍晦涩难懂。也正因如此，现在许多计算机网络专业的大学生，毕业后仍对这方面的知识一知半解，走上工作岗位后遇到实际的网络问题很难从原理方面分析出故障原因，更别说排除网络故障了。

虽然国内这方面的教材非常多，也不乏一些经典著作，但经笔者分析后认为大部分存在这样或那样的不足，要么通俗性较差，要么内容上过于浅显，更多的是照搬理论，难以使网络专业学生比较轻松地掌握全面、系统、专业的计算机网络原理和基础知识。但作为一名老的网络职业工作者和有着十几年计算机网络专业图书创作经验的老作者，深知这样一本看似非常基础，甚至有一些人认为非常简单的教材，要真正写出水平、写出权威并非易事，特别是在通俗性方面。现在许多书都把“零基础”当作卖点在宣传，但真正能做到零基础，并且在内容上有一定深度的书却并不多见。

另外，以前学计算机网络原理和网络基础知识的人可能大多数是计算机网络专业的学生，但随着计算机网络应用的普及，计算机网络知识几乎已成为所有 IT 专业必修的基础课程。而那些非计算机网络专业的学生对计算机网络可以说是真正的零基础，所以对这类教材在通俗性方面的要求会更高。要把那么深奥的计算机网络原理讲得能让这些零基础的读者理解和接受，难度就更是难以想象的了，这点笔者在创作过程中深有体会。尽管笔者在这方面也没有过深的造诣，但本着对信任、支持笔者的百万读者负责，怀抱着百万读者的期待和笔者自己二十多年的学习和工作经验积累，花了整整一年全职的创作时间写下了这本笔者认为在某些方面，特别是在通俗性方面有所提高的著作，力争使零基础的网络“菜鸟”也能轻松掌握复杂、深奥的计算机网络原理。希望这本书不会令广大读者朋友失望。

2. 时代的变迁：不懂计算机网络，你不敢说自己是 ITer

如果十年前你听到同行们都在说“不懂计算机，都不敢说自己是 ITer”，那么十年后的今天，你所听到的一定就是“不懂计算机网络，都不好意思说自己是 ITer”。更有人甚至会说“不懂计算机网络，就是现代文盲”。这些观点虽然可能有些偏颇，但也足以说明在全面信息化的今天，计算机网络在整个 IT 行业的重要性和基础性，它不再仅是网络专业人士必须掌握的，所有 ITer，甚至所有现代人都应该掌握。

以上虽然看似口号，但却实实在在地反映了当前整个 IT 行业都是以计算机网络作为中心和基础平台的这样一个现状。十年前，几乎所有的 IT 开发和应用都是以单一的计算机系统为平台的，几乎所有的计算机程序的运行环境都是单台计算机。十年后的今天，随着互联网接入的普及和宽带接入速度的提高，以及互联网和企业网络技术在应用上的普及与发展，一切都发生了变化。

过去单一的计算机系统平台根本无法满足当前无处不在、各种各样的网络应用需求，绝大多数 IT 开发和应用平台都转向了计算机网络这个无边的大平台。现在个人和企事业单位所进行的各项 IT 应用绝大多数都是基于计算机网络的，如浏览网页、收发邮件、写博客、写微博、网上购物、网上看电影 / 电视、网上玩游戏、网上听音乐、网络电子商务、网络营销、企业网络远程互联、网络会议、网络直播等。似乎我们现在所做的一切一切都离不开计算机网络，计算机网络成了实实在在的 IT 计算中心和基础应用平台。

现在基于单一计算机系统的应用已非常少了，且随着云计算、物联网这样的新型网络技术的应用和普及，可以十分清楚地预见，计算机网络这个平台才是整个 IT 行业发展的根本。就连现在我们仍然基于单机操作的办公应用软件，在不久的将来都可能全由云计算服务提供商通过互联网集中提供，再加上迅猛发展的移动互联网，计算机网络的基础地位将得到进一步巩固。到那时，如果连何为计算机网络都不懂，简单的计算机网络故障排除还要求助于人，这不就是现代文盲吗？你还敢说你是 ITer 吗？

3. 职业的挑战：计算机网络基础原理，网络职业发展的真正瓶颈

形势摆在我们所有 ITer 面前，但国内的现状却不怎么令人满意。先且不说所有 IT 行业，就是专门从事网络管理，或者网络工程行业的网络管理员和网络工程师，在计算机基础原理方面能比较深入地说出个一、二、三来的也没多少，碰到一个网络故障能从原理上进行全面分析的人更是少之又少，至少我所了解的是这样。可能有些人会说，会配置和管理网络不就行了？他们认为那些深奥的基础理论没什么用。其实说这样话的人还是不是很懂得网络管理和网络工程的真正职责，不是很理解这些网络基础原理的本质和重要性。网络管理的主要职责就是维护，在出现了网络故障时能快速、准确地排除故障，网络工程的主要职责就是为用户设计一个实用、符合各项标准，且稳定的系统。

很难想象，一个网络基础理论不扎实的网络管理人员如何能快速、准确地进行网络故障分析和排除，网络工程人员又如何能设计出一个符合标准、符合用户应用需求且能长时间保持稳定的系统。对于从事各种网络应用程序开发的程序员们来说，网络基础理论同样非常重要，一个不懂得网络体系结构，以及各层功能实现原理和应用接口的程序员，怎么可能设计出一个符合对应网络应用标准的应用程序？又怎么可能被用户认同？

随着计算机网络应用的不断高速发展，随着一大批快速成长型中小企业的高速发展，相信在不久的将来，全国将有无数企事业单位急需高水平、全面掌握基于计算机网络基础平台的 IT 设计和 IT 管理专业人才，到那时必将是一场残酷的职业竞争。如果连计算机网络基础原理都没有比较好的掌握，在起跑线上就输了，还如何参与竞争？

基于以上分析，我们可以十分清楚地知道，要成为一名合格的 IT 专业人才，无论你是从事 IT 应用开发，还是从事网络管理和网络工程设计，计算机网络基础原理都将是你的必修课！不要让自己输在起跑线上！

读者对象

本书内容看似非常专业、深奥，但这方面的知识现在已成为所有 ITer 的必修课。本书适合以下读者阅读，每类读者都可以通过阅读本书获得相应的收益：

- 所有想从事网络管理、网络工程设计的准网络管理员、准网络工程师
- 所有在网络职业发展道路遇到基础理论瓶颈的在职网络专业人士
- 所有大中专院校的 IT 专业学生

❑ 所有想学习计算机网络技术的“菜鸟”

如何阅读本书

本书虽然在知识讲解上已力争尽可能通俗化，但里面的知识点毕竟相当专业，所以在阅读本书时，建议注意以下几个方面：

❑ 不要刻意追求阅读速度

书中有的专业技术原理还是比较复杂的，建议大家在阅读时要一章、一节地消化，不要刻意追求阅读速度。一定要静下心来，认真阅读，千万别一目十行。

❑ 结合书中的示例阅读

本书所介绍的每项技术原理都会结合一些类比、演示示例，阅读者一定要仔细阅读示例讲解的每一步，最好自己也跟着示例进行计算、分析，以加深对原理的理解。

❑ 坚持，坚持，再坚持

尽管书中在通俗化方面已有较好的体现，但书中的内容毕竟全是基础理论，仍不能完全克服枯燥性，远不如图形操作界面那么简单明了，所以在阅读本书时一定要坚持，要静下心来学习，千万别半途而废。

❑ 选学内容可先不学

本书第 1 章和第 9 章属选学内容，主要是为已有一些基础的读者而准备的，所以如果你基础不是太好，可先跳过这两章，等以后有兴趣时再来学习。

本书的特色

“要登堂入室，必须先打开一扇门，或者一扇窗”，本书就是你登入计算机网络神圣殿堂的那扇门或窗。与同类图书对比，本书具有以下特色：

❑ 通俗易懂

这是本书最大的特色。为了能把复杂的技术原理讲得通俗易懂，书中不仅使用了大量的现实生活事例作为说明性的比喻，还列举了许多实例。同时，本书近 600 幅插图、近 100 个表格，可以帮助读者朋友更加直观地分析和理解各种复杂的技术原理。这是国内其他同类图书所没有的。

❑ 全面系统

本书的内容应该是同类图书中内容最全面、最系统的，不仅讲了目前主流的 TCP/IP 体系结构中的相关技术，还同时兼顾了 OIS/RM 和局域网体系结构的相关技术。更重要的是还包括了与各层对应的一些主要网络基础知识。真正做到“一本在手，网络无忧”。

❑ 专业深入

本书对所写到的每一部分内容都从专业角度进行了非常深入的剖析，使读者朋友不会有

在阅读其他同类图书时所有的许多重要知识点都“一笔带过”的感觉。笔者在写作之初就确立了“绝不一笔带过那些重要的知识点”的目标。

❑ 条理清楚

本书无论是从各章节内容安排上，还是从各小节内容组织上，条理都是比较清楚的。在本书中，对于一些比较复杂的内容都分出了多个小标题，重点突出，这样可以使读者朋友更加清楚地理解所介绍的内容，不会有整页或者几页都找不到主题、抓不住重点的现象。

勘误和支持

本书由王达主笔并统稿，参加编写、校验和排版的人员还有：何艳辉、王珂、沈芝兰、马平、何江林、刘凤竹、卢京华、周志雄、洪武、高平复、周建辉、孔平、尚宝宏、姚学军、张磊、刘学、李翔、王娇、李敏、吴鹏飞、宋希岭、刘中洲、潘朝阳、刘伟、曾平辉、李京杨、张跃、周平辉、王新宇、王薄、韩大为、宋宝强、史鹏宇、陆伟等。笔者在此对以上各位老师一并表示最由衷的谢意！尽管我们花了大量时间和精力校验，但由于水平有限，书中难免存在一些错误和瑕疵，敬请各位批评指正，万分感谢！

另外本书读者可以通过以下渠道享受相关服务：

❑ 多个专家博客和认证微博

笔者的主要博客：http://winda.blog.51cto.com、http://blog.csdn.net/lycb_gz、http://blog.chinaunix.net/uid/10659021.html。每个博客里面都有数百篇各方面的专业技术和职业指导文章，以及大量我以前所出版的图书的精彩试读文章。读者朋友不仅可以在里面学习各方面的知识，还可以直接向笔者提问。

笔者的两个微博：weibo.com/winda（新浪微博）、t.qq.com/winda2010（腾讯微博）。

❑ 超级 QQ 读者群

为方便全国各地读者交流，专门为本书读者新建了一个超大型、可容纳 2000 人的超级 QQ 读者群：196652938。由于读者众多，请尽快购买、加入，否则可能很快就没有位子了（加入时请注明本书名称）。

❑ 授课 PPT 免费下载

为了支持高校和培训机构老师讲课，本书为各位老师提供了授课 PPT，需要的朋友可以在机械工业出版社华章公司官网（www.hzbook.com）上下载，也可直接与笔联系获取（QQ：93220994，邮箱：lycb_gz@vip.sina.com）。

致谢

本书是笔者与机械工业出版社合作出版的第一部图书，感谢机械工业出版社华章公司，

以及杨福川老师给予我的这次十分难得的合作机会。由于本书内容较多，专业性较高，出版时间又非常紧，所以特别要感谢杨福川老师专门为本书抽调的精干编辑力量，及他在本书上线前做的大量推广工作；感谢孙海亮等其他所有编辑老师对本书的辛勤付出，我经常发现他们加班加点在编辑这部图书。期待本书能取得好的成绩，也期待通过此部图书合作的成功，为笔者与机械工业出版社展开更广泛的合作打下坚实基础。

王　达

目　录

前言

第 1 章　数制与编码 / 1

1.1　数制概述 / 2

1.1.1　常见数制类型及表示方法 / 2

1.1.2　不同数制之间的对应关系 / 3

1.2　不同数制间的相互转换 / 4

1.2.1　非十进制数转换成十进制数　/ 4

1.2.2　十进制数转换成非十进制数 / 6

1.2.3　非十进制数之间的相互转换 / 9

1.3　二进制数运算 / 10

1.3.1　二进制四则算术运算 / 11

1.3.2　二进制逻辑运算 / 13

1.4　二进制数的表示形式 / 15

1.4.1　二进制数的真值和字长 / 15

1.4.2　二进制数的四种表示形式 / 16

1.4.3　补码的加减法运算 / 19

第 2 章　计算机网络概述 / 23

2.1　计算机网络概述 / 24

2.1.1 计算机网络的定义 / 24
2.1.2 计算机网络的发展历史 / 25
2.1.3 计算机网络的基本组成 / 32
2.1.4 计算机网络的主要应用 / 34
2.2 计算机网络的分类 / 36
2.2.1 按网络所覆盖的地理范围分 / 37
2.2.2 按网络管理模式分 / 39
2.2.3 按传输方式分 / 43
2.3 计算机网络拓扑结构 / 44
2.3.1 网络拓扑结构相关基本概念 / 44
2.3.2 星型拓扑结构 / 45
2.3.3 环形拓扑结构 / 49
2.3.4 总线型拓扑结构 / 54
2.3.5 树形拓扑结构 / 59
2.3.6 网状拓扑结构 / 60
2.3.7 混合型拓扑结构 / 62
2.3.8 无线局域网的两种拓扑结构 / 64
第 3 章 计算机网络体系结构 / 66
3.1 典型计算机网络体系结构 / 67
3.1.1 OSI/RM 体系结构 / 67
3.1.2 TCP/IP 协议体系结构 / 70
3.1.3 局域网体系结构 / 71
3.1.4 例说网络体系结构各层主要功能 / 73
3.1.5 OSI/RM 和 TCP/IP 协议体系结构的比较 / 75
3.2 计算机网络体系结构通信原理 / 77
3.2.1 网络体系结构的数据通信原理 / 77
3.2.2 网络体系结构的对等通信原理 / 79
3.3 网络体系结构的设计考虑 / 82
3.3.1 网络体系结构中的层次划分依据 / 82
3.3.2 网络体系结构分层的好处 / 85
3.4 网络体系结构中的通信协议 / 86
3.4.1 理解计算机网络通信协议 / 86
3.4.2 网络通信协议的三要素 / 87

第 4 章　物理层 / 89
4.1　物理层概述 / 90
4.1.1　物理层的主要作用 / 90
4.1.2　物理层所定义的特性 / 91
4.2　数据通信基础 / 97
4.2.1　通信子网与资源子网 / 97
4.2.2　数据通信系统基本模型 / 98
4.2.3　数据通信的几个基本概念 / 99
4.2.4　数据传输类型 / 101
4.2.5　数据传输方式 / 105
4.2.6　数据传输模式 / 106
4.2.7　数据通信方式 / 108
4.3　数据传输速率与信道带宽 / 111
4.3.1　传输速率与信道带宽的基本概念 / 111
4.3.2　数字信号不失真传输的最大传输速率限制 / 112
4.3.3　模拟信号不失真还原的最小采样频率限制 / 114
4.4　数字基带信号编码 / 115
4.4.1　矩形脉冲数字信号基本波形 / 116
4.4.2　数字基带信号的传输码型 / 119
4.5　信号调制与解调 / 125
4.5.1　调制与解调的关键术语 / 125
4.5.2　ASK 调制与解调 / 127
4.5.3　FSK 调制与解调 / 130
4.5.4　PSK 调制与解调 / 135
4.6　物理层传输介质 / 140
4.6.1　导向性传输介质 / 141
4.6.2　光纤结构及主要附件 / 147
4.6.3　非导向介质 / 151
4.7　信道多路复用技术 / 152
4.7.1　频分复用及其原理 / 152
4.7.2　时分复用及其原理 / 154
4.7.3　波分复用及其原理 / 156
4.8　物理层接口 / 158
4.8.1　串行接口标准 / 158

4.8.2 RS-232 串行接口标准 / 159
4.8.3 其他 EIA 标准接口 / 163
4.8.4 X.21、X.24、X.36 和 EIA-530 接口规范 / 165
第 5 章 数据链路层 / 169
5.1 数据链路层基础 / 170
5.1.1 划分数据链路层的必要性 / 170
5.1.2 数据链路层结构 / 172
5.2 数据链路层主要功能及实现原理 / 175
5.2.1 数据链路管理 / 175
5.2.2 数据帧封装和透明传输 / 177
5.2.3 差错控制 / 180
5.2.4 流量控制 / 182
5.3 差错控制方案 / 183
5.3.1 奇偶校验码检错方案 / 183
5.3.2 循环冗余校验检错方案 / 185
5.3.3 反馈检测法 / 187
5.3.4 空闲重发请求方案 / 188
5.3.5 连续重发请求方案 / 190
5.3.6 海明纠错码 / 194
5.4 流量控制 / 198
5.4.1 XON/XOFF 流量控制方案 / 198
5.4.2 滑动窗口机制 / 199
5.5 面向字符的 BSC 协议 / 202
5.5.1 BSC 控制字符和数据块结构 / 202
5.5.2 BSC 协议数据透明传输原理 / 204
5.6 面向比特的 SDLC 和 HDLC 协议 / 205
5.6.1 HDLC 链路结构和操作方式 / 206
5.6.2 SDLC/HDLC 帧结构 / 207
5.6.3 SDLC/HDLC 帧类型及其标识方法 / 210
5.7 面向字符的 PPP 同步传输协议 / 212
5.7.1 PPP 简介 / 212
5.7.2 PPP 帧结构和透明传输原理 / 213
5.7.3 PPP 链路建立、使用和拆除流程 / 215
5.7.4 PPP 的 PAP/CHAP 身份认证 / 216

5.8 数据链路层主要网络设备 / 218
5.8.1 计算机网卡 / 218
5.8.2 网桥及其工作原理 / 221
5.8.3 二层交换机概述 / 224
5.8.4 二层交换原理 / 228
第 6 章 介质访问控制子层 / 231
6.1 MAC 子层基础 / 232
6.1.1 两种信道类型 / 232
6.1.2 MAC 子层概述 / 234
6.1.3 介质争用综述 / 235
6.2 CSMA 介质访问控制原理 / 237
6.2.1 非 – 坚持算法 / 237
6.2.2 1– 坚持算法 / 238
6.2.3 P– 坚持算法 / 239
6.3 CSMA/CD 介质访问控制原理 / 240
6.3.1 CSMA/CD 原理综述 / 241
6.3.2 冲突检测原理 / 242
6.3.3 冲突避让原理 / 243
6.3.4 CSMA/CD 的不足 / 245
6.4 局域网标准及以太网帧格式 / 246
6.4.1 IEEE 802 系列局域网标准 / 246
6.4.2 以太网帧格式综述 / 247
6.4.3 以太网 LLC 帧头部格式 / 251
6.4.4 以太网 SNAP 头部格式 / 251
6.4.5 以太网 MAC 帧 / 253
6.5 标准以太网规范及体系结构 / 255
6.5.1 标准以太网规范 / 255
6.5.2 标准以太网物理层结构 / 256
6.6 快速以太网规范及体系结构 / 258
6.6.1 快速以太网规范 / 259
6.6.2 快速以太网物理层结构 / 263
6.7 千兆以太网规范及体系结构 / 264
6.7.1 千兆以太网规范 / 264
6.7.2 1000Base-T 以太网技术 / 267

6.7.3 IEEE 千兆以太网物理层结构 / 269
6.8 万兆以太网规范及体系结构 / 270
6.8.1 万兆以太网规范 / 270
6.8.2 万兆以太网的物理层结构 / 273
6.9 IEEE 802.1d 协议 / 274
6.9.1 理解“网络环路”/ 274
6.9.2 STP 简介 / 275
6.9.3 STP 的基本工作原理 / 276
6.9.4 STP 的不足和增强技术 / 278
6.10 IEEE 802.1q 协议 / 279
6.10.1 划分 VLAN 的目的 / 279
6.10.2 理解 VLAN 的形成和工作原理 / 280
6.10.3 IEEE 802.1q 帧头部格式 / 282
6.11 IEEE 802.1w 协议 / 284
6.12 IEEE 802.1s 协议 / 286
6.12.1 MSTP 简介 / 286
6.12.2 MST 区域及工作原理 / 289
6.13 IEEE 802.1x 协议 / 291
6.13.1 IEEE 802.1x 认证设备角色 / 291
6.13.2 IEEE 802.1x 主机模式 / 292
6.13.3 IEEE 802.1x 认证流程 / 294
6.14 主要 WLAN 标准与技术 / 297
6.14.1 IEEE 802.11b 规范主要特性 / 298
6.14.2 IEEE 802.11a 规范主要特性 / 301
6.14.3 IEEE 802.11g 规范主要特性 / 303
6.14.4 IEEE 802.11n 规范主要特性 / 304
6.14.5 两个未正式发布的新规范简介 / 305
6.14.6 其他主要 WLAN 规范 / 306
6.14.7 WLAN MAC 帧格式 / 308
第 7 章 网络层 / 311
7.1 网络层概述 / 312
7.1.1 划分网络层的必要性 / 312
7.1.2 网络层主要作用 / 314
7.2 网络层数据交换及相关技术 / 315

7.2.1 线路交换 / 316
7.2.2 存储－转发 / 317
7.2.3 虚电路分组交换 / 320
7.2.4 数据报分组交换 / 322
7.2.5 虚电路交换和数据报交换的比较 / 323
7.3 网络层协议及报文格式 / 324
7.3.1 IP 协议基本功能 / 325
7.3.2 IPv4 的不足 / 326
7.3.3 IPv6 的主要优势 / 327
7.3.4 IPv4 数据报头部格式 / 328
7.3.5 IPv6 数据报头部格式 / 332
7.3.6 IPv6 扩展报头 / 335
7.3.7 IPv4 数据报的封装与解封装 / 336
7.3.8 IPv4 数据报的分段与重组 / 338
7.3.9 ARP 协议报文格式及 ARP 表 / 339
7.3.10 ARP 地址解析原理 / 341
7.3.11 ICMP 协议及报文格式 / 342
7.3.12 IPv6 协议簇中的其他协议 / 345
7.4 路由和路由算法 / 347
7.4.1 路由的分类 / 348
7.4.2 路由算法基础 / 352
7.4.3 路由表基础 / 355
7.4.4 路由优先级 / 356
7.4.5 路由算法设计目标和设计考虑 / 357
7.5 几种主要的路由算法解析 / 359
7.5.1 最短路径路由算法 / 359
7.5.2 扩散算法 / 362
7.5.3 距离矢量路由算法 / 363
7.5.4 链路状态路由算法 / 367
7.6 网络拥塞控制方法和原理 / 371
7.6.1 网络拥塞控制方法 / 371
7.6.2 死锁及其预防 / 374
7.7 网络层设备及主要技术 / 376
7.7.1 路由器主要硬件技术 / 376

7.7.2 路由器主要软件技术 / 381
7.7.3 三层交换机 / 385
7.7.4 三层交换机硬件结构 / 386
7.7.5 三层交换原理 / 387
7.7.6 三层交换示例 / 389
7.7.7 三层交换机和路由器的主要区别 / 391
第 8 章 IP 地址和子网 / 393
8.1 IPv4 地址 / 394
8.1.1 IPv4 地址基本格式 / 394
8.1.2 子网掩码 / 395
8.1.3 IPv4 地址的基本分类 / 396
8.1.4 有类 / 无类 IPv4 网络 / 400
8.1.5 网络地址、主机地址和广播地址 / 402
8.1.6 IPv4 地址前缀表示形式 / 404
8.1.7 几种特殊的 IPv4 地址 / 405
8.2 IPv4 子网划分与聚合 / 407
8.2.1 VLSM 子网划分的基本思想 / 407
8.2.2 全 0 子网与全 1 子网 / 408
8.2.3 VLSM 子网划分方法 / 409
8.2.4 VLSM 子网划分示例 / 410
8.2.5 子网聚合方法及示例 / 413
8.3 IPv4 NAT 基础 / 415
8.3.1 NAT 的主要应用 / 416
8.3.2 与 NAT 相关的主要术语 / 416
8.3.3 NAT 地址基本转换原理 / 419
8.3.4 NAT 类型 / 420
8.4 IPv6 地址基础 / 422
8.4.1 IPv6 地址表示形式 / 422
8.4.2 IPv6 地址中的二进制数与十六进制转换 / 424
8.5 IPv6 地址类型 / 425
8.5.1 IPv6 单播地址 / 426
8.5.2 IPv6 组播地址 / 430
8.5.3 IPv6 任播地址 / 431
8.5.4 IPv6 主机和路由器地址 / 432

8.5.5 IPv6 地址前缀表示形式 / 433
8.6 IPv6 地址自动配置 / 434
8.6.1 IPv6 地址自动配置的类型 / 434
8.6.2 自动配置过程 / 435
第 9 章 路由协议及工作原理 / 437
9.1 RIP 路由协议 / 438
9.1.1 RIP 路由度量机制 / 438
9.1.2 RIP 路由更新机制 / 440
9.1.3 RIP 路由收敛机制 / 442
9.1.4 RIP 报文格式 / 445
9.2 OSPF 路由协议 / 446
9.2.1 OSPF 协议简介 / 446
9.2.2 OSPF 的 AS 与 Area / 448
9.2.3 OSPF 网络路由器类型 / 449
9.2.4 DR 和 BDR / 450
9.2.5 OSPF LSA 类型 / 452
9.2.6 Backbone（骨干）区域 / 454
9.2.7 Stub（末梢）区域 / 455
9.2.8 Totally Stub 区域和 NSSA 区域 / 456
9.2.9 OSPF 路由计算基本过程 / 458
9.2.10 OSPF 报头格式 / 460
9.3 IS-IS 路由协议 / 464
9.3.1 ISO 网络基础 / 464
9.3.2 IS-IS 路由协议基本术语 / 465
9.3.3 IS-IS 路由及路由器类型 / 468
9.3.4 IS-IS 与 OSPF 区域及路由器邻接关系比较 / 469
9.3.5 IS-IS PDU 报头格式 / 472
9.3.6 IIH PDU 包格式 / 473
9.3.7 LSP PDU 包格式 / 475
9.3.8 SNP PDU 包格式 / 476
9.3.9 IS-IS PDU 可变字段格式 / 477
9.3.10 IS-IS 的两种地址格式 / 478
9.3.11 IS-IS 与 OSPF 的比较 / 480
9.3.12 IS-IS 最短路径计算和路由表生成原理 / 481

9.4 BGP / 483
9.4.1 BGP 概述 / 483
9.4.2 BGP AS / 484
9.4.3 BGP 地址簇模型 / 486
9.4.4 BGP speaker 和 peer 的关系 / 488
9.4.5 BGP peer 会话建立 / 490
9.4.6 BGP 的路由属性 / 490
9.4.7 BGP 的消息类型及报文格式 / 494
第 10 章 传输层 / 498
10.1 传输层概述 / 499
10.1.1 划分传输层的必要性 / 499
10.1.2 传输层的端到端传输服务 / 501
10.1.3 传输层服务 / 502
10.1.4 TSAP 和 TPDU / 504
10.1.5 传输连接建立阶段的主要 TPDU / 507
10.1.6 数据传输阶段的主要 TPDU / 508
10.1.7 传输连接释放阶段的 TPDU / 512
10.1.8 传输服务原语 / 513
10.2 传输层服务功能 / 517
10.2.1 传输层寻址方案 / 517
10.2.2 传输连接建立 / 520
10.2.3 重复传输连接的解决方法 / 521
10.2.4 数据传输 / 524
10.2.5 传输连接释放 / 525
10.2.6 流量控制 / 526
10.2.7 多路复用 / 529
10.2.8 崩溃恢复 / 529
10.3 TCP 概述 / 530
10.3.1 TCP 的主要特性 / 530
10.3.2 TCP 数据段格式 / 531
10.3.3 TCP 套接字 / 534
10.3.4 TCP 端口 / 537
10.3.5 TCP 连接的状态转移 / 539
10.3.6 TCP 传输连接的建立 / 542

10.3.7 TCP 传输连接的释放 / 544
10.4 TCP 的可靠传输 / 546
10.4.1 TCP 的数据段确认机制 / 547
10.4.2 TCP 的超时重传机制 / 549
10.4.3 TCP 的选择性确认机制 / 550
10.5 TCP 的流量控制 / 552
10.5.1 TCP 的流量控制简介 / 552
10.5.2 基于传输效率的考虑 / 554
10.6 TCP 的拥塞控制 / 555
10.6.1 TCP 拥塞控制简介 / 555
10.6.2 TCP 拥塞控制方案 / 557
10.7 UDP 概述 / 560
10.7.1 UDP 的基础知识 / 560
10.7.2 UDP 数据报头部格式 / 561
第 11 章 应用层 / 563
11.1 应用层概述 / 564
11.1.1 应用层组件及典型应用服务 / 564
11.1.2 应用层的 C/S 服务模型 / 565
11.2 Web 服务基础 / 566
11.2.1 Web 服务模型 / 566
11.2.2 万维网的全球统一标识 / 567
11.2.3 万维网文档标记 / 569
11.2.4 HTML 文档类型 / 570
11.2.5 HTML 文档的“三超属性”/ 572
11.2.6 HTTP 服务访问基本流程 / 573
11.2.7 HTTP 的主要特性 / 574
11.2.8 HTTP 请求报文格式 / 575
11.2.9 HTTP 响应报文格式 / 577
11.3 DNS 服务 / 579
11.3.1 DNS 技术的引入背景 / 580
11.3.2 DNS 命名方案的设计思想 / 582
11.3.3 DNS 名称空间 / 583
11.3.4 DNS 名称服务器 / 586
11.3.5 DNS 报文格式 / 589

11.3.6 DNS 数据传输方式 / 593
11.3.7 DNS 递归解析原理 / 594
11.3.8 DNS 迭代解析原理 / 596
11.4 DHCP 服务 / 599
11.4.1 BOOTP 和 DHCP 简介 / 599
11.4.2 DHCP 服务的主要功能及应用环境 / 600
11.4.3 DHCP 报文及其格式 / 601
11.4.4 DHCP 服务的 IP 地址自动分配原理 / 604
11.4.5 DHCP 服务的 IP 地址租约更新原理 / 611
11.4.6 DHCP 中继代理服务 / 611
11.5 电子邮件服务 / 615
11.5.1 电子邮件系统的基本结构 / 615
11.5.2 电子邮件消息格式 / 617
11.5.3 SMTP 请求命令和应答消息 / 619
11.5.4 SMTP 服务的工作原理 / 623
11.5.5 POP3 请求命令及应答消息 / 626
11.5.6 POP3 服务的工作原理 / 628
11.5.7 IMAP4 简介 / 630

第 5 章

数据链路层

要在一条通信线路上传送数据，除了必须建立一条物理线路（物理层的功能）之外，还必须有一些规程或协议来控制这些数据的传输，以保证被传输数据的正确性。实现这些规程或协议的硬件和软件加上物理线路就构成了本章要介绍的“数据链路层”（Data Link Layer，DLL）。

我们知道，物理层中也有许多规程或协议，但它们是用来构建物理传输线路、建立物理意义的网络通信，不是用来控制数据传输的。设计数据链路层的主要目的就是在原始的、有差错的物理传输线路的基础上，采取差错检测、差错控制与流量控制等方法，将有差错的物理线路改进成逻辑上无差错的数据链路，以便向它的上一层——网络层提供高质量的服务。就像我们修好了路，还得制定一些交通法规，使路上行驶的车辆必须按照一定的规则行驶，否则可能会经常出现交通事故。这些“交通法规”也为了使这些车辆到达某个车站（这里所说的“车站”就相当于计算机网络体系结构中的“网络层”）时能有序进、出站，最终使这条数据通信之“路”发挥它本来的作用。

在不同网络体系结构中，数据链路层的结构和所包括的功能并不完全一样。本章主要针对广域网中的数据链路层和局域网体系结构中的逻辑链路控制（LLC）子层的功能及相关技术进行全面、深入的介绍。读者要着重要掌握的是数据链路层的链路管理、数据帧封装、差错控制、流量控制这几项功能的实现原理，BSC、SDLC、HDLC 和 PPP 这几种典型的数据链路层协议及二交换原理。有关局域网体系结构中的媒介访问控制（MAC）子层的功能和相关技术将在下章介绍。

5.1 数据链路层基础

在所有计算机网络体系结构中都直接或间接地包含了数据链路层（在 TCP/IP 协议体系结构中数据链路层的功能包含在网络访问层中）。数据链路层和它下面的物理层其实本质作用都是一样的，就是用来构建进行网络通信、访问的通道，只不过物理层构建的是一条物理通道，而数据链路层构建的是真正用于数据传输的逻辑通道。正因如此，在目前 Internet 中广泛使用的 TCP/IP 协议体系结构中，物理层和数据链路层是集中划分在网络访问层这一层之中的。

5.1.1 划分数据链路层的必要性

虽然说物理层和数据链路层的本质作用都是用来构建网络通信、访问通道，但它们所建立的通信通道是不一样的。首先要说明的一点是，在物理层上构建的是物理链路，在数据链路层上构建的是逻辑链路或者数据链路，它们是不同的概念，但确实有许多读者分不清楚。

物理链路是指在物理层设备（包括传输介质、物理接口和收发器等）和相应物理层通信规程作用下形成的**物理线路，是永久存在的**，且是不可删除的（除非物理拆除）；逻辑链路则是通信双方在需要进行数据通信时，在数据链路层设备和相应的通信规程作用下建立的**逻辑链路，可以是永远存在的**（如局域网中的以太网链路），**也可以不是永久存在的**（如广域网中的链路），是否永久存在要视具体的数据链路层服务类型而定。这里还有一个“链路”的概念，它是指相邻节点之间的那段数据线路。

在看到物理链路和逻辑链路之间区别的同时，又要看到它们之间的联系，那就是逻辑链路必须建立在物理链路之上。如果通信双方的物理线路都不通，是不可能建立用于数据传输的逻辑链路的。我们可以这样来理解它们之间的关系，物理链路是基础线路，相当于一条公路的路基，而逻辑链路是在物理链路之上的高级线路，可以理解为在公路上铺设了柏油或者水泥的车道。它们之间的关系如图 5-1 所示。

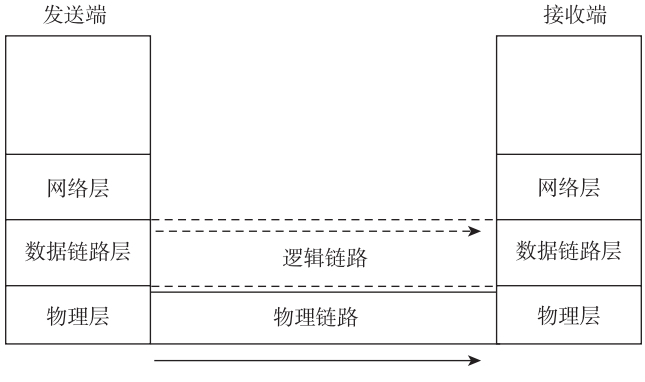


图 5-1 “物理链路”和“逻辑链路”的关系

无论在多么复杂的网络中，从逻辑意义上来讲，真正的数据传输通道就是数据链路层中所定义的数据链路，只不过在要经过多个网络的数据通信中，数据链路是分段的，每个网络都有一段链路，这些链路段连接起来就是整个数据通信的数据链路。图 5-2 所示的是一个有三个网段经过两个路由器（路由器 A 和路由器 B）相连的网络，现假设 PC 用户 A 要向 PC 用户 B 发送一个数据，它的实际数据传输过程如图 5-2 中各网络物理层之间的实线箭头所示，而逻辑上可以等同各部分数据链路层之间构建的“逻辑链路”之间的数据转发，如图 5-2 中虚线箭头所示。

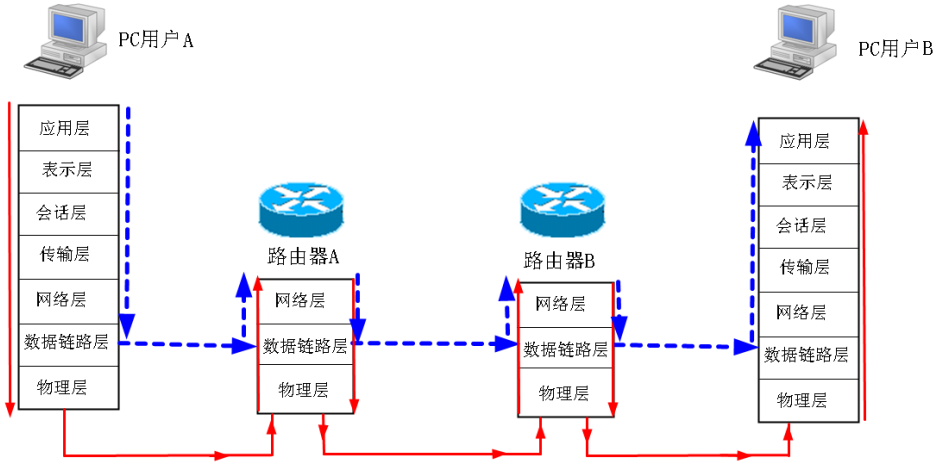


图 5-2　数据在网络中的传输方向

说到这里，可能有的读者会问，既然在“物理层”中构建了数据传输通道（也就是物理层中的“信道”），那为什么还要多加一个“数据链路层”的功能呢？其实这主要有两方面的原因：

- 一是由于物理层传输介质的多样性，通信规程也各不相同，性能不稳定，而数据链路层中构建的逻辑链路不考虑不同物理链路上传输介质及其通信规程上的区别（也就是我们通常所说的数据链路层可以屏蔽物理层中传输介质的不同），只是从逻辑意义上构建一条性能稳定、不受传输介质类型影响的逻辑数据传输通道。就像修建公路时，所用的材质也可能不一样，有的用普通的泥巴，有的用沙石，还有的用大石材。如果仅靠一些基础材料修建公路，可能修好的公路通车性能很差，有的甚至根本不能通车，只能步行。但如果我们在这些公路上再统一铺一层钢筋混凝土，那么这些由不同材料修建的公路就可能满足基本相同的通车性能的要求了。
- 再一个原因是，在物理层中数据是一位位地单独传输的，不仅数据传输效率低下，而且容易出现数据传输差错（如出现某些数据位丢失或者错位，在物理层中又没有相应的通信规程进行数据传输差错控制），就像一条不能通车的普通公路上，人只能一个

个地步行，还可能出现迷路现象一样。而在数据链路层中数据是以“帧”为单位进行传输的，一个帧通常是有数千个比特位的，不仅传输效率提高，还不容易出错（因为在数据链路层中有专门的通信规程来负责数据传输差错控制），就像在能通车的公路上以车为单位运载人一样，不仅传输效率提高，还不容易出现各种交通事故。至于数据链路层的主要功能，将在本章后具体介绍。

5.1.2 数据链路层结构

在正式介绍“数据链路层”主要功能和实现原理前，我们先要明白，各种计算机网络体系结构中，数据链路层的结构是不完全一样的。在 OSI/RM 和 TCP/IP 体系结构中，数据链路层就一层，而在局域网体系结构中是可细分为两个子层的，那就是逻辑链路控制（Logical Link Control，LLC）子层和介质访问控制（Medium Access Control，MAC）子层，如图 5-3 所示。

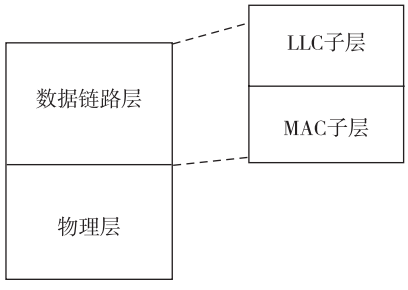


图 5-3 局域网体系结构“数据链路层”的两个子层

经验之谈 设置 MAC 子层的目的主要是解决多路通信的介质争用和寻址，所以其主要适用于广播型链路和点对多点链路。对于点对点链路来说，没什么太大意义，因为点对点链路不存在介质争用，是一路通信所独占的。具体将在下章介绍。

从图 5-3 可以看出，在数据链路层中，与它的下一层——物理层相邻的是 MAC 子层，与它的上一层——网络层相邻的是 LLC 子层。所以 MAC 子层接受物理层的服务，为 LLC 子层服务，而 LLC 子层则是接受 MAC 子层服务，为网络层服务。而各层（其他层也一样）之间接受服务或者提供服务的地方就是 SAP（Service Access Point，服务访问点）。下面先来了解什么是 SAP。

1. 各层的 SAP

从 SAP 的中文名称“服务访问点”可以看出，它就是上层访问相邻下层所提供服务的点。我们知道，在计算机体系结构中，下层是为相邻的上层提供服务的，而下层对它的所有上层都是透明的。也就是上层不会具体管它的下面各层是如何工作的，只需要它的相邻下层提供必要的服务即可。

SAP 是邻层实体（“实体”也就是对应层的逻辑功能）间实现相互通信的逻辑接口，位于两层边界处。从物理层开始，每一层都向上层提供服务访问点（应用层除外），每一层都有 SAP，但不同层的 SAP 内容和表示形式都是不一样的。各层 SAP 的表示形式是对应层第一个单词的第一个字母加上“SAP”，如物理层 SAP 表示为 PSAP（Physical layer Service Access Point），对应的就是网络通信中设备的具体物理接口；数据链路层 SAP 表示为 DLSAP（Data Link Control layer Service Access Point），对应的就是各个物理接口的 MAC 地

址；网络层 SAP 表示为 NSAP（Network layer Service Access Point），对应的就是各物理接口上配置的网络地址（如 IP 地址，但在 OSI/RM 中网络地址不一定是 IP 地址，要视具体的网络层协议而定，如还可以是 IPX 地址）；传输层 SAP 表示为 TSAP（Transport layer Service Access Point），对应的就是具体网络应用通信所用的传输层端口；会话层 SAP 表示为 SLSAP（Session layer Service Access Point），对应的就是具体网络应用会话进程；表示层 SAP 表示为 PLSAP（Presentation layer Service Access Point），对应的就是具体网络应用进程中的用户标识。

从以上介绍可以得知，其实 SAP 每层所对应的“地址”，但是针对一个具体的网络通信（注意，这里特别说明一下不是**数据通信**）来说，不同层中的 SAP 数是不一样的。如物理层的 PSAP 只有一个（就是对应的物理接口），数据链路层的 DLSAP 也只有一个（就是对应物理接口的 MAC 地址），在网络层中虽然每个物理接口可以有多个 IP 地址，但是对于一个具体的数据通信来说，它也只有一个，所以 NSAP 也只有一个，传输层及以上各层的 SAP 就可以有多个了，因为每一个网络通信中可以同时进行多路网络应用（当前有多少个网络应用进程，就需要多少个 SAP），实现多路**数据通信**。正因如此，针对一个具体的网络通信中各层的 SAP 可以描述为图 5-4 所示的形式。

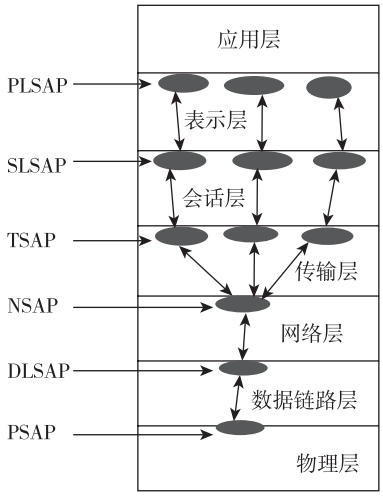


图 5-4　OSI/RM 各层的 SAP

2. MAC 子层

从“MAC”的中文名称“介质访问控制”可以看出，MAC 子层的最基本功能就是如何控制不同用户数据传输中对物理层传输介质的访问，其中包括介质访问时的寻址（这里是通过 MAC 地址进行的），以及解决可能发生的介质访问冲突（也就是我们通常听到的“仲裁介质的使用权”，即规定站点何时可以使用通信介质）。如 IEEE 802.3 以太网标准 MAC 子层规范了如何在总线型网络结构下使用传输介质；IEEE 802.4 令牌总线（Token-Bus）标准 MAC 子层规范了如何在总线的网络结构下利用令牌（token）控制传输介质的使用；IEEE 802.5 令牌环（Token-Ring）标准 MAC 子层规范了如何在环状网络结构下利用令牌来控制传输介质的使用；IEEE 802.11 标准的无线局域网标准 MAC 子层规范了如何在无线局域网络的结构下控制传输介质的使用。

具体而言，数据链路层中与各种传输介质访问**有关**的问题都放在“MAC 子层”来解决。其主要功能包括：数据帧的封装 / 卸装，帧的寻址和识别，帧的接收与发送，帧的差错控制、介质访问冲突控制等。有关 MAC 子层的具体功能和技术介绍将在下章进行。

3. LLC 子层

从“LLC”的中文名称“逻辑链路控制”可以看出，LLC 子层的最基本功能就是负责数

据链路层中逻辑链路（逻辑链路就是物理层信道中的物理链路在通过 LLC 子层协议作用后形成的虚拟链路）的控制，其中包括逻辑链路的建立和释放，控制信号交换、数据流量控制，解释上层通信协议传来的命令并且产生响应，以及克服数据在传送的过程当中所可能发生的种种问题，如数据发生错误、重复收到相同的数据、接收数据的顺序与传送的顺序不一致等。在 LLC 子层方面，IEEE 802 系列标准中只制定了一种标准——IEEE 802.2，各种不同局域网都使用相同的 LLC 子层通信标准。

由于网络层上可能有许多种通信协议同时存在，而且每一种通信协议又可能同时与多个对象沟通，因此当 LLC 子层从 MAC 子层收到一个数据包时必须能够判断要送给网络层的是哪一个通信协议。为了达到这种功能，在 LLC 子层中提供了“数据链路层”的 SAP，作为与“网络层”通信交互的接口（每路通信需要一个 SAP 接口，如图 5-5 所示）。为了能够辨认出 LLC 子层上传送的数据从哪里来，要到哪里去，在 LLC 子层上传输的每个 LLC 数据单元（LLC Protocol Data Unit，LPDU）上都会有“目的服务访问点”（Destination Service Access Point，DSAP）和“源服务访问点”（Source Service Access Point，SSAP）这两个地址。具体的 LPDU 帧格式将在本章后面介绍。

在计算机网络中进行的数据传输，虽然实际上是从发送端的高层一路经过数据链路层、物理层，然后再从接收端的物理层、数据链路层一直传输到对应的高层（如图 5-5 中实线箭头所示），而从逻辑意义看，数据是从发送端数据链路层到接收端数据链路层间的一段段逻辑链路上进行传输的（如图 5-5 中虚线箭头所示），因为在物理层中传输的比特流最终还是要转换成数据帧在数据链路层中传输。

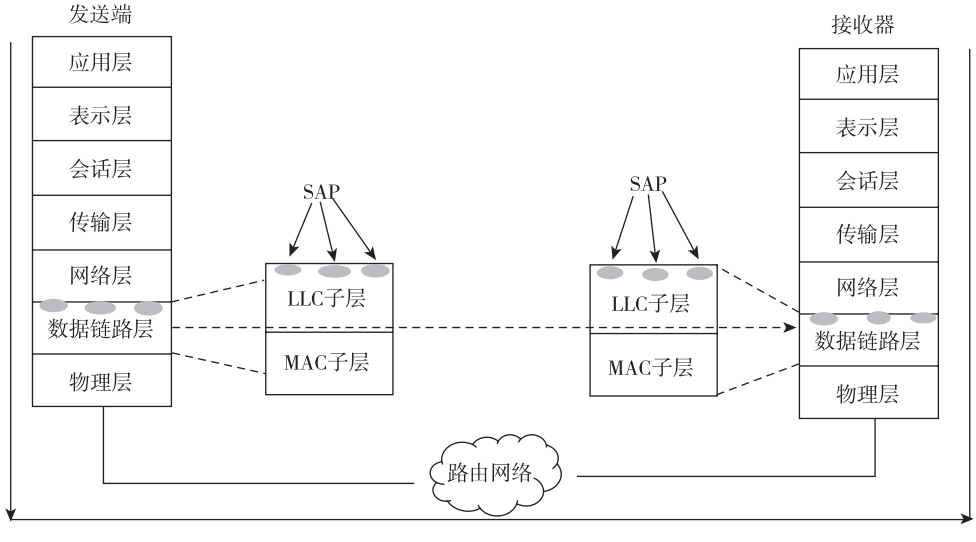


图 5-5 LLC 子层的 SAP 及数据传输原理示例

具体而言，数据链路层中与传输介质访问**无关**的问题都集中在 LLC 子层来解决，为网

络层提供服务。 其主要功能包括逻辑链路的建立和释放、提供与网络层的接口（也就是前面说到的 SAP）、数据传输差错控制、给数据帧加上传输序列号等。

5.2　数据链路层主要功能及实现原理

数据链路层位于网络体系结构中“网络层”（在 TCP/IP 协议体系结构中称网际互连层）的下层，所以它的一项基本功能就是向网络层提供透明、可靠的数据传输服务（在计算机网络体系结构中，下一层是为相邻的上一层服务的）。“透明”是指要使在数据链路层上所传输的数据在内容、格式及编码上都没有限制，也就是要使一些本来用于特殊用途的控制字符（具体有哪些控制字符将在本章后面介绍）也能像正常的数据一样传输，使接收端不要误认为这些字符为控制字符；可靠的传输使数据从发送端无差错地在数据链路上传输到目的接收端。总体而言，数据链路层（其实这里主要是针对 LLC 子层）的主要功能就是四个方面：数据链路管理、封装成帧、透明传输、差错控制。下面具体介绍。

5.2.1　数据链路管理

在本章前面说到了，在数据链路层中要形成一条更有利于数据传输的数据链路，而不是直接利用下面物理层中建立的物理链路。物理链路在没有人为拆除时是永久存在的，而数据链路一般是非永久存在的（但局域网中的数据链路是永久的），仅当有数据传输时建立并存在，在数据传输完后自动拆除。数据链路是由数据链路层中的 LLC 子层通过相应的通信规程（也就是通常所说的协议）建立并管理的。

说明　数据链路分为点对点链路和点对多点链路（或“广播链路”）两种。点对点链路就是一个节点只与另一个节点连接起来的链路，用于建立点对点通信。它所采用的是点对点协议，如 PPP（点对点协议），PPPoE（基于以太网的点对点协议）。点对多点链路就是一个节点同时与多个节点连接建立起来的链路，用于建立点对多点通信。它所采用的通常是点对多点协议，如以太网协议、WLAN 协议，还有本章后面将要介绍的 HDLC（高级数据链路控制）协议。

---

1. 数据链路层提供的服务类型

根据数据链路层协议的不同，所建立的数据链路类型也会有不同。同时我们知道，数据链路层是为上面的网络层提供服务的，所以这些不同协议所建立的数据链路向网络层提供的服务类型也有所不同。总体上可把这些数据链路服务分为以下三类：有确认的面向连接服务、有确认的无连接服务、无确认的无连接服务。前者称为面向连接服务（Connection-oriented Servce），后面两者称为无连接服务（Connetionless Service）。

说明　其实不仅本章中所讲的数据链路层协议有面向连接和无连接两种服务类型，在网络层和传输层协议中也有这两种类型的，如网络层中的 X.25 协议是面向连接的，而 IP 协议

则是无连接的；传输层中的 TCP 协议是面向连接的，而 UDP 协议是无连接的。这些将在第 10 章进行介绍。

（1）有确认的面向连接服务

有确认的面向连接服务里面包括两层含义：一是在提供服务时，必须先建立好双方通信连接；二是在提供服务时，必须要求对方确认后才进行。这种服务类型存在三个阶段，即数据链路建立、数据传输、数据链路释放等阶段。举个现实中的例子，就像我们打电话，我们打电话给某个人时，首先就是要拿起电话拨号（相当于建立连接的过程），然后对方拿起电话，问一下看是不是打错了（这就是一个“确认”过程）。确认不是打错的电话后，双方开始通话，这就相当于在数据链路中进行数据传输的过程；通话完毕，双方挂掉电话，相当于链路释放的过程。

从以上这个打电话的例子可以看出，数据链路层中有确认的面向连接服务是独占链路的，只有在当前数据传输完成，释放了链路后，其他用户才可能与同一个接收端进行数据传输。就像你打电话给你的朋友时，其他人再打电话给你朋友听到的是忙音，只有等你结束了与你朋友的通话后，其他人才可以打通你那朋友的电话。同样，从以上分析可以得出，有确认的面向连接服务非常可靠，这一则是因为有专门的通信链路，在一路通信使用某条链路时，其他通信不能同时使用这条链路；再则是这种服务类型不会向错误的接收端进行数据传输，也可确认接收端正确地接收了发送来的数据，而且是按数据帧发送顺序接收，每一帧只接收了一次，因为它规定接收端在接收到每一个数据帧（每个帧都有编号）后必须对发送端进行确认，就像打电话一样，只有对方确认你是要找他的，他才可能接听你的电话。

大多数广域网中通信子网的数据链路层协议采用有确认的面向连接服务，如 SLIP（串行线路协议）、PPP（点对点协议）、PPPoE（基于以太网的点对点协议）、HDLC（高级数据链路控制）协议等。

（2）有确认的无连接服务

有确认的无连接服务与有确认的面向连接服务的相同之处就是接收端在接收到的每一个数据帧时都向发送端确认；不同之处在于它在进行数据传输前是不需要建立专门的数据链路的，自然也不需要在数据传输结束后释放数据链路（事实上是因为这类服务所用的数据链路已建立起来，而且是永久存在的，所以不用另外建立，如局域网中的链路）。就像我们从快递公司寄快递信件一样，信的投递路线我们不用管（事实上投递路线已经有了），但是在收件人收到信件时必须要求收件人签收（也就是要对接收到的每一个数据帧进行确认）。

有确认的无连接服务虽然不用建立专门的连接，但仍可以保证数据的可靠传输，因为它有“确认”功能，如令牌环网和令牌总线网中的数据传输就是采用这种服务类型，接收端在接收到一个数据帧时会发送确认信息给发送端的。这类服务的另外一个主要用途就是用于一些不可靠信道中的数据传输，如**各种无线通信系统**。

（3）无确认的无连接服务

无确认的无连接服务与前面的有确认的无连接服务的相同之处就在于它们都不需要在进行数据传输前先建立专门的数据链路，也就是无须先在通信双方建立通信连接；不同之处就是它在进行数据传输时不要求接收端对所接收到的每一个数据帧进行确认。就像我们从邮局寄平信一样，信件投递路线我们不用管（事实上投递路线已经有了），而且当信件到达收信人时，也不用收件人签名确认。

这种服务类型看似不可靠，但它是建立在可靠的通信线路基础之上的，所以数据传输仍然是非常可靠的。如我们常用的以太网中所使用的各种以太网协议就是采用这种服务的，因为以太网中的数据链路性能非常好，数据可靠传输有保障。在以太网中的数据链路始终是存在的，不用另外建立，在**以太网中进行数据传输时接收端也不用对接收到的每一帧进行确认**。

2. 数据链路管理

LLC 子层的链路管理功能主要是针对前面所介绍的有确认的面向连接服务类型（主要应用于广域网中）。它包括三个主要阶段：链路建立、链路保持、链路释放。

在这种数据链路层服务中，链路两端的节点要进行通信前，发送端的数据链路层必须先确认对方已处于就绪状态，并交换一些必要的信息以对帧序号进行初始化，然后双方才能建立连接；在传输过程中为个数据连接是要持续保持的；如果出现差错，需要重新初始化，重新自动建立连接。传输完毕后则要释放所占用的数据连接，以供其他通信所用。

数据连路层的这种链路连接的建立、维持和释放过程就是数据链路层的链路管理功能。在多个站点共享同一物理信道的情况下（例如在局域网中），如何在要求通信的站点间分配和管理信道也属于数据链路层管理的范畴。

5.2.2　数据帧封装和透明传输

我们知道数据链路层位于物理层和网络层之间。在发送端，数据链路层是接收来自网络层的数据分组，而在接收端它是接收来自物理层的比特流，所以数据链路层的成帧功能就包含两方面的含义：一是将来自网络层的数据分组封装成数据帧，二是将来自物理层的一个个比特流组装成数据帧。因为帧封装（将物理层比特流组装成帧时，称为帧同步）通常是与透明传输一起考虑并实现的，所以在此一并介绍。不过本节仅介绍其基本原理，在本章后面还会结合具体的数据链路层协议详细介绍这些帧封装和透明传输原理。

1. 数据包的帧封装原理

通过前面的学习我们就已经知道，网络层传输的包（packet，又称分组），在数据链路层中传输的是“帧”（frame）。数据包到达数据链路层后加上数据链路层的协议头和协议尾就构成了一个数据帧。在每个帧的前部加上一个帧头部，在帧的结尾处加上一个帧尾部，把网络层的数据包作为帧的数据部分，就构成了一个完整帧。帧头和帧尾就是作为帧的起始和结

束标志，也就是帧边界，如图 5-6 所示。

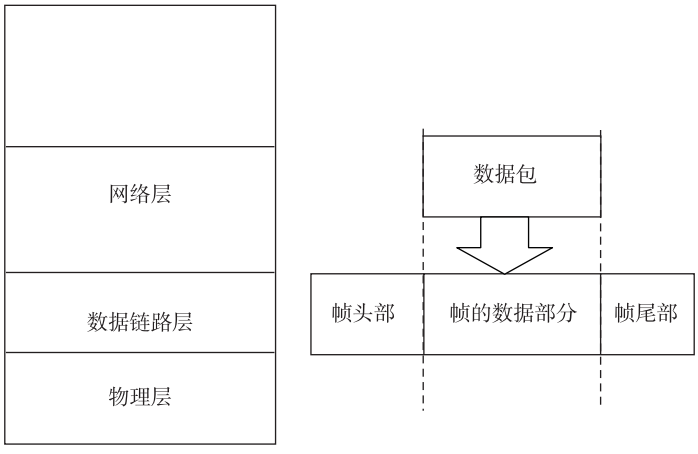


图 5-6　网络层数据包封装成帧的示意图

由数据包封装成的数据帧其大小是受对应的数据链路层协议的 MTU（最大传输单元）限制的，如以太网数据链路层封装网络层 IP 包的 MTU 值为 1500 字节（这是指帧中数据部分，也就是来自网络层整个数据分组，最大不能超过 1500 字节，但不包括帧头和帧尾部分）。同时，帧还有最小大小限制，如以太网帧中封装的 IP 包最小值为 46 字节，如果封装的 IP 包小于最小帧要求时，就要用一些特殊字符进行填充，以满足对应链路中传输最小帧的限制。

2. 比特流的帧组装及透明传输原理

在发送端数据链路层中的帧到达物理层后就会以比特位为单位进行传输，而不是以帧为单位进行传输。尽管在并行传输方式中，可以一次传输一个或多个字节，但每条线路中的传输单位还是比特位。发送端以比特位方式一位位地传输到接收端的物理层，然后接收端的物理层把比特流向数据链路层传输，到达后又要将比特流封装成数据帧，这就是数据链路层的帧组装方式了，其实也就是我们前面提到的帧同步问题。帧同步的目的就是要使接收端的数据链路层对从物理层传输而来的一串串比特流以帧为单位进行区分。

本节先简单介绍以下几种常用的帧同步方法的基本同步原理：字节计数法、字符填充的首尾定界符法、比特填充的首尾定界符法、违法编码法。本章的后面在介绍具体的数据链路层协议时还将具体介绍它们所采用的同步方法。

（1）字节计数法

这是一种以一个特殊字符代表一个帧的起始，并以一个专门的字段来标识当前帧内字节数的帧同步方法。接收端可以通过对该特殊字符的识别从比特流中区分出每个帧的起始，并从专门字段中获知每个帧后面跟随的“数据”（Data）字段的字节数，从而可确定出每个帧的结束位置。

这种面向字节计数的同步规程的典型实例是DEC公司的DDCMP（Digital Data Communications Message Protocol，数字数据通信报协议）。在DDCMP协议通信中，数据是在源站点与从站点之间以编号的数据消息的形式进行交换的，而从站点是以未编号的响应和控制消息的形式向主站点返回的。下面看看在这个协议的数据帧中如何实现帧同步，或者是如何成帧的：

在DDCMP协议帧格式（如图5-7所示）中，SOH字段是一个帧的帧头开始部分，有其固定的值（十进制为129，八进制值为201，十六进制值为81），就相当于一个帧开始的特殊字符；在NUM字段中为每个数据帧分配一个编号，从“1”开始，并以“1”为增量进行递增，最大值为256（也就是模为256），以确保在从站点中的正确消息序列，同时在COUNT字段中指出本数据帧中DATA字段的大小，这些都是用来进行帧同步的。

8	14	2	8	8	8	16	8 ～ 131064	16bits
SOH	COUNT	FLAGS	RESP	NUM	ADDR	BLKCK1	DATA	BLKCK2

图 5-7　DDCMP 帧格式

（2）字符填充的首尾定界符法

该同步方法是用一些特定的控制字符来定界一个帧的起始与结束，如IBM的BSC协议在每个数据块的头部用一个或多个同步字符“SYN”来标记数据块的开始；尾部用字符“ETX”来标记数据决的结束。图5-8所示的是要传输一个“ADFGJ”的字符串，在帧的头部加上了两个SYN控制字符，用于标识该帧的开始，在结束位置加了ETX控制字符，用于标识该帧的结束。

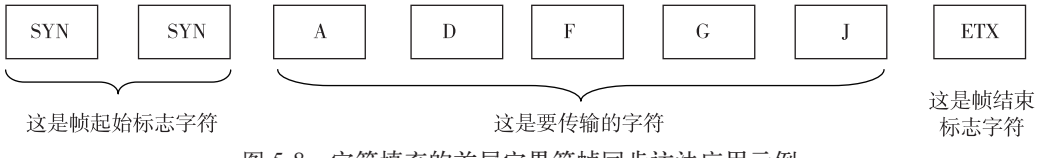


图 5-8　字符填充的首尾定界符帧同步访法应用示例

另外，为了不使数据信息中与以上特定字符相同的字符被误判为帧的首尾定界符，可以在这种数据帧的帧头填充一个转义控制字符（Data Link Escape-Start of TeXt，DLE），这就属于“透明传输”的范围了，这部分内容将在本章后面介绍具体的数据链路层协议时介绍。

（3）比特填充的首尾定界符法

该帧同步方法是通过在帧头和帧尾各插入一个特定的比特串（如01111110）来标识一个数据帧的起始与结束，这个帧头、帧尾特定比特串称为帧标志符。如传输的比特流为1001110101，组装成帧后就是01111110100111010101111110。

而为了透明传输，也就是为了避免在信息位中出现的与帧起始和结束标志符相似的比特串时被误判为帧的首、尾标志，采用了比特填充的方法。比如上面采用的特定模式

为“01111110”，则对信息位中的任何连续出现的 5 个“1”（因为帧标志符中是有 5 个连续“1”），发送端自动在其后插入一个“0”。 如要传输的数据帧为“0110111111011111001”（因为其中有 5 个连续的“1”，很可能被误认为是帧首、尾标志），采用比特填充后，实际传送的是 01111110011011111*0*10111111*0*001*0*1111110（注意，前、后面两个“01111110”是帧首、尾标志符）。另外，因为在原信息中，有一段比特流与帧标志符类似，为了与用于标识帧头和帧频尾的特定模式字符区别，在有 5 个连续“1”的比特位后面加插入一个“0”（斜体“0”）。而接受方在收到上述最终数据后进行发送端的逆操作，首先去掉两端的特定模式字符，然后在每收到连续 5 个“1”的比特位后自动删去其后所跟的“0”，以此恢复原始信息。

比特填充帧同步方式很容易由硬件来实现，性能优于字符填充方式。所有面向比特的同步控制协议均采用统一的帧格式，不论是数据，还是单独的控制信息均以帧为单位传送，其典型代表是 ISO 的 HDLC 协议，在它的首尾均有标志字段（Flag，8 位，即 0111110），具体将在本章后面介绍。

（4）违法编码法

该帧同步方法是在物理层采用特定的比特编码方法时采用。例如，曼彻斯特编码方法，将数据“1”编码成“高 − 低”电平对（在半个码元处跳变，下同，具体参见 4.4.2 节），将数据“0”编码成“低 − 高”电平对。而高 − 高电平对和低 − 低电平对在数据比特中是违法的，因此可以借用这些违法编码序列来定界帧的起始与终止。违法编码法不需要任何填充技术，便能实现数据的透明性，但它只适用于采用冗余编码的特殊编码环境。

5.2.3 差错控制

说明 因数据链路层中的差错控制功能涉及比较多且比较复杂的技术，所以在此仅进行综合介绍，具体的差错控制技术将在本章后面介绍。

在上节介绍的“成帧”功能解决了帧同步问题，也就是接收端可以区分每个数据帧的起始和结束了，但是还没有解决数据正确传输的两方面问题：一是如果有帧出现了错误怎么办？二是如果有帧丢失了怎么办？这都是数据链路层确保向网络层提供可靠数据传输服务要解决的问题，也就是数据链路层的差错控制功能。

要实现差错控制功能，就必须具备两种能力：一具备发现差错的能力，二是具备纠正错误的能力。就像我们要发行一个“问题”小孩一样，你首先要知道他的“问题”在哪里，也就是存在问题的根源在哪里，然后才能采取适当的方法纠正这个“问题”小孩身上的问题。

1. 差错检测

在数据链路层检测数据传输错误的方法一般是通过对差错编码进行校验来实现，常见的校验方法有奇偶校验码（Parity Check Code，PCC）、循环冗余校验（Cyclic Redundancy Check，CRC）两种。它们都统称为检错码（error-detecting code）。

奇偶校验码是一种校验代码传输正确性的方法，是根据被传输的一组二进制代码的数位中“1”的个数是奇数或偶数来进行校验的。采用“1”的奇数个校验的方法称为奇校验，反之称为偶校验，但采用何种校验是事先确定好的。具体做法是在传输的二进制代码最后专门设置一个奇偶校验位，用它使这组代码中 1 的个数为奇数或偶数（具体是偶数还是奇数，要视所采用的是偶校验还是奇校验），然后再在接收端进行校验，看里面的“1”的个数是否仍与原来一样的奇数或偶数，来确定数据传输的正确性。

循环冗余校验是一种根据传输或保存的数据而产生固定位数校验码的方法，主要用来检测或校验数据传输或者保存后可能出现的错误。生成的数字在传输或者储存之前计算出来并且附加到数据后面，然后接收端进行检验确定数据是否发生变化。

有关以上这两种差错检测方法在本章后面有专门的介绍。

2. 差错纠正

在差错纠正方面，数据链路层针对不同的传输类型采取了不同的纠错方法。对于面向字符的异步传输（如键盘与主机的通信、ATM 传输协议等）一般是采用“反馈检测”的方法来进行纠错。就是在接收端收完一帧数据后，向发送端发送回所接收到的完整数据帧，由发送端通过与原始发送的帧进行比较来判断接收端是否正确接收了对应帧。如果判断是出了错，则发送端向接收端发送一个 DEL 字符及相应的帧信息，提示接收端删除对应的帧，然后重发该帧；否则表示接收端已正确接收了对应的帧，不重发对应的帧。但对于在传输过程中完全丢失的数据就不能采用这种纠错方法了，因为接收端根本没有收到这帧数据，所以也就不会向发送端发回反馈信息，发送端自然就不能确认接收端是否正确接收了对应的帧。

为了解决这个问题，通常在数据发送时引入计时器（Timer）来限定接收端发回反馈信息的时间间隔。当发送端发送一帧数据的同时启动计时器，若在限定时间间隔内没有收到接收端的反馈信息，即计时器超时（Timeout），则可认为传的对应帧已出错，或丢失，继而发送端知道要重新发送对应的数据帧。同时，为了避免同一帧数据可能被多次重复发送，引发接收端多次收到同一帧并将其递交给网络层的危险，采用对发送的帧进行编号的方法，即同一个帧的编号是一样的，从而使接收端能从帧编号来区分是新发送来的帧，还是已经接收但又重新发送来的帧，以此来确定要不要将重新接收到的帧递交给网络层。

由于在“反馈检测法”中一帧数据会在信道中至少往返传输两次，传输效率低，所以一般不采用这种差错控制方法，而是采用一种称为“自动重发请求”（ARQ 法）的方法。实现原理就是先让发送端将要发送的数据帧附加一定的冗余检错码（如前面介绍的 PCC、CRC 码）一起发送，接收端则根据检错码对数据帧进行错误检测，若发现错误，就返回请求重发的响应（不用返回全部的帧），发送端收到请求重发的响应后，便重新传送该数据帧。

另外，还有一些编码本身具有自动纠正错误的能力，称为“纠错码”（Error-correcting Code），在数据链路层中常用的如海明码（Hamming Code）。

以上差错控制方法都将在本章后面具体介绍。

5.2.4 流量控制

“流量控制”包括两方面的含义：一是发送端的数据发送速度与接收端的数据接收速度要匹配，否则接收端来不及接收就会造成数据在传输过程中的丢失。这个很好理解，比如几个人站成一排传递货物时，如果中间有个人速度比较慢，而上面的人不断传来货物，肯定就会把来不及接收的货物放在地上（不进入正常的传递之中）。二是发送端的数据发送速度要与线路上的承载速率（与线路信道带宽有关）相匹配，否则也会造成数据在传输过程中的丢失。这个也很好理解，就像一条小河，上游来的水量很大，超过了小河所能承载的能力，这时上游来的水肯定就不会有原来那么大的流速（形成速率瓶颈），甚至可能会漫过小河河堤而流到外面。

注意 流量控制并不是数据链路层所特有的功能，许多高层协议中也提供流量控功能，只不过流量控制的对象不同而已。比如，对于数据链路层来说控制的是相邻两节点之间数据链路上的流量，而对于传输层来说控制的是从源到最终目的端之间的流量。

在数据通信中，由于通信双方各自使用的设备工作速率和缓冲存储的空间的差异，可能出现发送端发送能力大于接收端接收能力的现象，如若此时不对发送端的发送速率作适当的限制，在接收端前面来不及接收的帧将被后面不断发送来的帧所“淹没”，从而造成帧的丢失。由此可见，数据链路层上的“流量控制”功能实际上是对发送端数据传输速率的控制，使其数据发送速率不超过接收端所能承受的数据接收能力。考虑到在接收端还需要对来自物理层的比特流进行一系列的处理，如帧封装，向发送端发送返回确认帧等，所以通常是要使发送端的发送速率略小于接收端的数据处理能力。

在数据链路层中进行流量控制的方案有两种：一是基于反馈的流量控制方案，二是基于速率的流量控制方案。

1. 基于反馈的流量控制方案

“基于反馈的流量控制方案”就是接收端在接收到一个数据帧后，要向发送端发送一个确认帧，表示发送端可以继续向它发送数据了。这种方案也称“停－等”方案（也就是将在5.3.4 节介绍的“空闲重发请求”方案），就是发送端在发送一帧数据后必须等待接收端返回确认响应消息，然后才能发送下一数据帧。接收端是通过检查帧的校验序列（FCS），无错则发送确认帧，否则不发送返回消息，表示该帧已出错，要求重发。

其实这也就是利用上节介绍的差错控制方案。但这里存在一个问题，就是在数据帧或确认帧丢失时，双方会无休止等待。解决这一问题的方法是发送后使用定时器，在发送端发送一帧数据时会启动这个定时器，要求接收端必须在这个时间内返回确认帧，否则就认为这个数据帧已丢失，重发该数据帧。当然，与上节介绍的差错控制方案时提到的一样，这也可能造成接收端接收到多个一样的数据帧，解决这一问题的方法就是给每个帧加上一个编号，这样编号一样的帧接收端只需接收其中一个即可，其他的丢弃。

由此可以看出，“停－等”方案其实就是直接利用了上节介绍的纠错方案，这里提到的几种措施可以说是一环扣一环，最终可以既确保数据的可靠传输（也就是“差错控制”功能），又有效地控制了双方通信的流量（也就是“流量控制”功能）。

2. 基于速率的流量控制方案

“基于速率的流量控制方案”是基于窗口滑动机制的速率控制方案，它规定发送端一次可以发送多少个数据帧，限制了发送端的数据传输速率，而无须接收端发回确认帧。这种机制等在本章后面也有专门介绍。其实它与第 10 章要介绍的传输层流量控制方案是极其类似的，不同的只是数据链路层的流量控制是针对链路两端的点对点控制，而传输层则是直接针对通信双方的端到端系统。

5.3　差错控制方案

前面已说到，数据链路层的差错控制功能分为差错检测功能和差错纠正功能。在差错检测方面通常在数据帧中加上一些具有检错功能的冗余代码（称为检错码，Error-detecting Code），然后根据这些冗余代码所表达的信息，以及接收端对数据帧的奇偶性重新计算的结果就可以发现数据帧在传输过程中是否出现差错。如前面说到的 PCC（奇偶校验码）、CRC（循环冗余校验）都属于检错码。但它们均不具有自动纠错功能，不能确定是哪一个或哪一些位出错了，也不能纠正这些差错，通常也是与纠错方案中的自动请求重发法（ARQ 法）结合起来进行差错控制的。

在差错纠正方面主要有反馈检测法和自动重发请求法两种（自动重发请求法中又分为几种，如空闲重发请求法，连续重发请求法等）。也有一些具有自动纠错功能的编码，称为纠错码（Error-correcting Code），又称前向纠错码（Forward Error Correction），如海明码（Hamming Code）。这些方案都将在下面各小节中具体介绍。

5.3.1　奇偶校验码检错方案

奇偶校验码（PCC）是奇校验码和偶校验码的统称，是一种有效检测**单个错误**的检错方法。它的基本校验思想是在原信息代码的最后添加一位用于奇校验或偶校验的代码，这样最终的帧代码是由 $n-1$ 位信元码和 1 位校验码组成，可以表示成为（n，$n-1$）。加上校验码的最终目的就是要让传输的帧中“1”的个数固定为奇数（采用奇校验时）或偶数（采用偶校验时），然后通过接收端对接收到的帧中“1”的个数的实际计算结果与所选定的校验方式进行比较，就可以得出对应帧数据在传输过程中是否出错了。如果是奇校验码，在附加上一个校验码以后，码长为 n 的码中“1”的个数为奇数；如果是偶校验码，则在附加上一个校验码以后，码长为 n 的码中“1”的个数为偶数（0 个“1”也看成是偶数个“1”）。奇偶校验方法可以以通过电路来实现，也可以通过软件来实现，在此我们只要知道它校验的原理即可。

下面打个很简单的比喻来说明 PCC 的校验原理。现假设一学校要组织各班学生参加一

次长跑运动，同时有几名外地学生想参加长跑运动，需在每班插入一个外地学生，但插入的学生性别不能随意，最终要求插入外地学生后各班的女学生数必须为偶数（相当于采用偶校验方式，当然也可以要求必须为奇数，若为奇数，则此时为奇校验方式），这样就可确定每班插入的这位外地学生的性别了。如果长跑后各班清点人数时，发现有些班的女生数量不是原来规定的偶数（或奇数），则证明这些班的学生在长跑过程中走乱了（相当于数据在传输途中出现了差错）或者有学生掉队（相当于数据在传输途中丢失了）了。这就是我们这里所说的奇偶校验原理。

我们知道 ASCII 字符是 8 位编码，其中高 7 位是信息代码，最后 1 位就是奇偶校验位。如现在传输的是 ASCII 字符，如果采用奇校验，则每个 ASCII 字符代码中“1”的个数均必须为奇数个，如果发现某个字符中“1”的个数是偶数，则这个 ASCII 字符在传输过程中肯定出错了；同理，如果采用的是偶校验，则每个 ASCII 字符代码中“1”的个数均必须为偶数个，如果发现某个字符中“1”的个数是奇数，则这个 ASCII 字符在传输过程中也肯定出错了。其他采用奇偶校验方式的数据的校验原理也是一样的。

假设现在要传输一个 ASCII 字符，它的高 7 位代码为 1011010，现在要采用奇校验方法，则该字符的校验码为“1”，整个 ASCII 字符代码就是 1011010 1，因为该字符中高 7 位信息代码中的“1”的个数是偶数个（4 个），必须再加一个“1”才能为奇数；同理，如要采用偶校验方法，则该字符的校验码为“0”，整个 ASCII 字符代码就是 1011010 0，因为该字符中高 7 位信息代码中的“1”的个数已是偶数个（4 个），所以最后一位中不能再是“1”，只能为“0”。

这里要注意，无论是采用奇校验，还是偶校验，每一信息的校验位是“0”还是“1”都是不固定的，要视信息前 $n-1$ 位中“1”的具体个数而定。如果采用的是“奇校验”，则要求所传输的 n 位信息中“1”的总个数必须为奇数：如果前面 $n-1$ 位中“1”的个数已是奇数，则第 n 位的校验码位一定是“0”，只有这样才能确保整个 n 位信息中“1”的个数仍为奇数；如果前面 $n-1$ 位中“1”的个数是偶数，则第 n 位的校验码位一定是“1”，只有这样才能确保整个 n 位信息中“1”的个数为奇数。

如果采用的是偶校验，则要求所传输的 n 位信息中“1”的总个数必须为偶数（0 个也看成是偶数个）：如果前面 $n-1$ 位中“1”的个数是奇数，则第 n 位的校验码位一定是“1”，只有这样才能确保整个 n 位信息中“1”的个数为偶数；如果前面 $n-1$ 位中“1”的个数是偶数，则第 n 位的校验码位一定是“0”，只有这样才能确保整个 n 位信息中“1”的个数仍为奇数。

另一个要注意的是，奇偶校验方法只可以用来检查单个码元错误，检错能力较差，所以一般只用于本身误码率较低的环境，如用于以太局域网中、用于磁盘的数据存储中等。如现在采用的是奇校验方法，传输的整个 8 位 ASCII 字符代码为 10100100（最低位为校验位），如果里面前 7 位信息代码有一位出现了差错，由原来的“0”变为“1”，或者由原来的“1”变为“0”，都会改变里面“1”的个数，最终导致与对应的奇校验方法不一致，在这种情况

下 PCC 能判断这个字符传输出错。但是如果里面前 7 位信息代码同时有 2 位或多位出现了差错，最终的结果可能会使字符的整个 8 位代码中“1”的个数奇偶性不变，而不能判断这个字符传输出错。如以上字符代码变为 10001100，结果“1”的个数仍为奇数个（3 个），但实际上该数据已不是原来的数据了。这就为什么奇偶校验方法只能检测出单个码元错误的原因。

作为课后练习，大家自己再分析一下，如果所传输的二进制序列是 11101100101，现要采用奇校验，则校验位的值是什么？采用偶校验呢？

5.3.2　循环冗余校验检错方案

上节介绍的奇偶校验码（PCC）只能校验一位错误，本节要介绍的循环冗余校验码（CRC）的检错能力更强，可以检出多位错误。

1. CRC 校验原理

CRC 校验原理看起来比较复杂、难懂，因为大多数书上基本上都是以二进制的多项式形式来说明的。其实其原理很简单，根本思想就是先在要发送的帧后面附加一个数（这个数就是用来校验的校验码，但要注意，这里的数也是二进制序列的，下同），生成一个新帧发送给接收端。当然，这个附加的数不是随意的，它要使所生成的新帧能与发送端和接收端共同选定的某个特定数整除（注意，这里不是直接采用二进制除法，而是采用一种称为模 2 除法的方法）。到达接收端后，再把接收到的新帧除以（同样采用模 2 除法）这个选定的除数。因为在发送端发送数据帧之前就已附加了一个数，做了去余处理（也就已经能整除了），所以结果应该没有余数。如果有余数，则表明该帧在传输过程中出现了差错。

说明　模 2 除法与算术除法类似，但它既不向上位借位，也不比较除数和被除数的相同位数值的大小，只以相同位数进行相除。模 2 加法运算为：1+1=0，0+1=1，0+0=0，无进位，也无借位。模 2 减法运算为：1−1=0，0−1=1，1−0=1，0−0=0，也无进位，无借位，相当于二进制中的逻辑异或运算。也就是比较后，两者对应位相同则结果为“0”，不同则结果为“1”。如 100101 除以 1110，结果得到商为 11，余数为 1，如图 5-9a 图所示。再如 11×11=101，如图 5-9b 图所示。

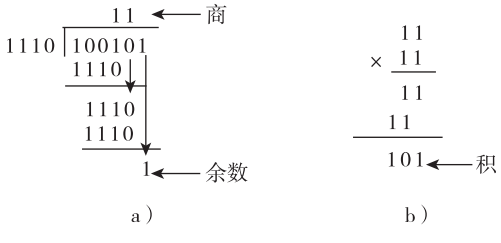


图 5-9　“模 2 除法”和“模 2 乘法”示例

具体来说，CRC 校验的实现分为以下几个步骤：

1）先选择（可以随机选择，也可按标准选择，具体在后面介绍）一个用于在接收端进行校验时，对接收的帧进行除法运算的除数（是二进制比较特串，通常是以多项方式表示，所以 CRC 又称多项式编码方法，这个多项式又称生成多项式）。

2）看所选定的除数二进制位数（假设为 k 位），然后在要发送的数据帧（假设为 m 位）后面加上 $k-1$ 位“0”，接着以这个加了 $k-1$ 个“0“的新帧（一共是 $m+k-1$ 位）以“模 2 除法”方式除以上面这个除数，所得到的余数（也是二进制的比特串）就是该帧的 CRC 校验码，又称 FCS（帧校验序列）。但要注意的是，**余数的位数比除数位数只能少一位，哪怕前面位是 0，甚至是全为 0（附带好整除时）也都不能省略**。

3）再把这个校验码附加在原数据帧（就是 m 位的帧，注意不是在后面形成的 $m+k-1$ 位的帧）后面，构建一个新帧发送到接收端，最后在接收端再把这个新帧以“模 2 除法”方式除以前面选择的除数，如果没有余数，则表明该帧在传输过程中没出错，否则出现了差错。

通过以上介绍，大家一定可以理解 CRC 校验的原理了。

从上面可以看出，CRC 校验中有两个关键点：一是要预先确定一个发送端和接收端都用来作为除数的二进制比特串（或多项式）；二是把原始帧与上面选定的除进行二进制除法运算，计算出 FCS。前者可以随机选择，也可按国际上通行的标准选择，但**最高位和最低位必须均为“1”**，如在 IBM 的 SDLC（同步数据链路控制）规程中使用 CRC—16（也就是这个除数一共是 17 位）生成多项式 $g(x)= x^{16} + x^{15} + x^2 +1$（对应二进制比特串为 11000000000000101）；而在 ISO HDLC（高级数据链路控制）规程、ITU 的 SDLC、X.25、V.34、V.41、V.42 等中使用 CCITT—16 生成多项式 $g(x)= x^{16} + x^{15} + x^5 +1$（对应二进制比特串为 11000000000100001）。

2. CRC 校验码的计算示例

由以上分析可知，除数是随机或者按标准选定的，所以 CRC 校验的关键是如何求出余数，也就是校验码（CRC 校验码）。

下面以一个例子来具体说明整个过程。现假设选择的 CRC 生成多项式为 $G(X) = X^4 + X^3 + 1$，要求出二进制序列 10110011 的 CRC 校验码。下面是具体的计算过程：

1）首先把生成多项式转换成二进制数，由 $G(X) = X^4 + X^3 + 1$ 可以知道，它一共是 5 位（总位数等于最高位的幂次加 1，即 4+1=5），然后根据多项式各项的含义（多项式只列出二进制值为 1 的位，也就是这个二进制的第 4 位、第 3 位、第 0 位的二进制均为 1，其他位为 0）很快就可得到它的二进制比特串为 11001。

2）因为生成多项式的位数为 5，根据前面的介绍得知，CRC 校验码的位数为 4（校验码的位数比生成多项式的位数少 1）。因为原数据帧 10110011，在它后面再加 4 个 0，得到 101100110000，然后把这个数以“模 2 除法”方式除以生成多项式，得到的结果为 0100，如图 5-10 所示。注意参考前面介绍的“模 2 除法”运算法则。

3）用上步计算得到的 CRC 校验替换帧 101100110000 后面的 4 个“0”，得到新的帧

101100110100。再把这个新帧发送到接收端。

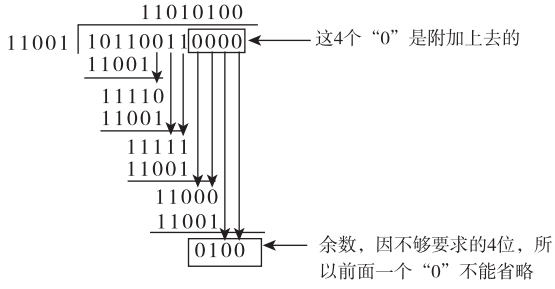


图 5-10　CRC 校验码计算示例

4）当以上新帧到达接收端后，接收端会把这个新帧再用上面选定的除数 11001 以“模 2 除法”方式去除，验证余数是否为 0，如果为 0，则证明该帧数据在传输过程中没有出现差错，否则出现了差错。

通过以上对 CRC 校验原理的剖析和 CRC 校验码计算示例的介绍，大家应该对这种看似很复杂的 CRC 校验原理和计算方法比较清楚了。

下面大家做一个练习：假设 CRC 生成多项式为 $G(X) = X^5 + X^4 + X + 1$，要发送的二进制序列为 100101110，求 CRC 校验码是多少？

5.3.3　反馈检测法

“反馈检测法”是一种最简单、最容易实现，但并不常用的差错控制方法。这种差错控制方法的信道利用率低，因为在这种差错控制方法中每个数据帧均要求至少在信道中往返传输两次。下面具体介绍。

1. 反馈检测法基本原理

反馈检测法不需要利用前面介绍的 PCC（奇偶校验码）和 CRC（循环冗余校验码）检测错码技术，其把差错检测和差错纠正的任务全部交给发送端。它要求接受端在接收到每一个数据帧后均要向发送端发送一个表示是否接收了该数据帧的反馈信息，且这个反馈信息就是原来由发送端发给接收端的原始数据帧。发送端在收到接收端发送的反馈信息后，通过对比保存在缓存中原来该帧的数据来判断接收端是否正确接收了该数据帧。

这种要求接收端必须向发送端反馈的做法，有些类似于我们向某人发送一个需要对方确认的重要邮件一样，如果收件人收到了你发送的这封邮件，要求对方对你进行回复确认；如果对方收到的邮件已遭破坏，也可以向你反映，然后你可以重发这封邮件给他，直到对方收到了正确的邮件。但也不完全与上述过程等同，因为这里的邮件反馈并不需要发回原来邮件的全部内容，只是一条表示收到邮件了的反馈信息。

如果经过比较发现，接收端反馈来的某数据帧与原始帧不一样，则表明对应帧在传输过程中出现了差错，接收端接收到的对应帧是错误的，则接收端向发送端发送一个 DEL 字符

和相应的帧编号（下面将介绍到），通知接收端删除对应的帧，由发送端重发对应的数据帧，直到接收到接收端反馈来的数据对应帧与原来发送的该帧数据完全一样为止；如果经比较接收端返回来的某帧与原来发送的对应帧完全一样，则表示接收端已正确接收到该数据帧，不再重发，继续发送下一数据帧。

2. 两项附加技术

从以上反馈检测原理可以发现，这里有一个问题，那就是可能因一些原因导致一些数据帧在传输过程中完全丢失了，接收端根本没收到这些数据帧，自然也就不会向发送端发送对应数据帧的反馈信息了。就像你要求收件人在收到你发送的邮件时给你回信，但对方根本没收到你的邮件，自然也就不会有回信了，你还总在那里等他的回信。

为了避免出现这种情况，通常引入计时器（Timer）来限定接收端发回反馈消息的时间，即当发送端发送一帧数据时即自动启动计时器，如果在计时器中限定的时间内仍没有收到接收端发来的反馈信息，则可认为该帧的数据传输出现了差错或丢失，就主动重新发送该帧。就像你对某人发送邮件前就告诉对方，“我现在就发送邮件给你，你必须在 2 小时内回复我是否收到了这封邮件，否则我认为你没收到这封邮件，会重新发一封给你”。

以上计时器方案虽然解决了由于数据丢失而引发的意外，但可能由于网络线路比较忙，送达的某帧数据在到达接收端时有所延迟，超出了计时器限定的时间范围，发送端仍然会重发对应的数据帧，这样就可能使接收端多次接收同一数据帧了。为了防止发生这种现象，在“数据链路层”又采取了“帧编号”（也就是“帧序列号”）方法对发送的每个数据帧进行编号（同一数据帧编号一样，无论发送多少次），从而使接收端能从该编号来区分是新发来的帧，还是已经接受但又重新发来的帧，从而确定要不要将接收到的帧递交给网络层。“数据链路层”通过使用“计数器”和“帧编号”两种措施来保证每个数据帧最终都能被正确地递交给目标网络层一次。

5.3.4 空闲重发请求方案

从上节介绍的“反馈检测法”差错控制原理中可以看出，其虽然简单，容易实现，也有较高的可靠性，但每个数据帧实际上在信道中均被传输了两次，造成信道利用率降低。也正如此，这种差错控制方法一般用于面向字符的异步传输中，因为在这种情形下，传输的数据量比较小，信道传输效率并不是主要问题。

实用的差错控制方法，既要求传输可靠性高，又要求信道利用率高。为此可使发送端将要发送的数据帧附加一定的冗余检错码（如 PPC、CRC 等）一并发送，接收端则根据检错码对数据帧进行差错检测；若发现错误，就返回请求重发的响应，发送端收到请求重发的响应后，便重新传送该数据帧。这种差错控制方法就称为自动重发请求（Automatic Repeat reQuest，ARQ）法。

ARQ 差错控制方案仅需返回少量控制信息（接收端不必重传整个数据帧），便可有效地确认所发数据帧是否正确被接收。ARQ 法有几种具体的实现方案，空闲重发请求（Idle RQ）

和“连续重发请求”（Continuous RQ）是其中最基本的两种方案。本节先介绍“空闲重发请求”差错控制方案，下节将介绍连续重发请求差错控制方案。

空闲重发请求方案又称停－等（Stop and Wait）法，该方案规定发送端每发送一帧后就要停下来，然后等待接收端发来的确认信息（这就是停－等的意思），仅当接收端确认（ACK）信息后才继续发送下一数据帧。如果收到的是否认（NAK）消息，表示接收端接收的数据有错，请求发送端重发。另外，在计时器超时时，发送端也会重发对应的帧。

图 5-11a、b 所示分别是正确接收数据时的数据帧发送流程和接收到有错误数据时的数据帧发送流程。

空闲重发请求差错控制方案的具体实现过程如下：

1）发送端每次仅将当前数据帧作为待确认帧保留在缓冲存储器中，当发送端开始发送数据帧（如图 5-11 所示的 data0）时，随即启动计时器。

2）当接收端收到这个数据帧时，先利用帧中附带的检错码进行校验，确认无差错后，即向发送端返回一个确认信息（如图 5-11a 所示的 ACK0、ACK1 和图 5-11b 所示的 ACK0）；当检测到该帧有错误时，向发送端返回一个否认帧（如图 5-11b 所示的 NAK0），同时丢弃该帧。

3）如果发送端在计时器中规定的时间内收到来自接收端的确认信息，即将计时器清零，清除缓存中的待确认帧，然后才开始下一数据帧（如图 5-11a 所示的 data1）的发送；若发送端在规定时间内未收到来自接收端的确认信息（即计时器超时），则重发存放于缓冲器中的待确认数据帧（如图 5-11b 所示的 data1）。

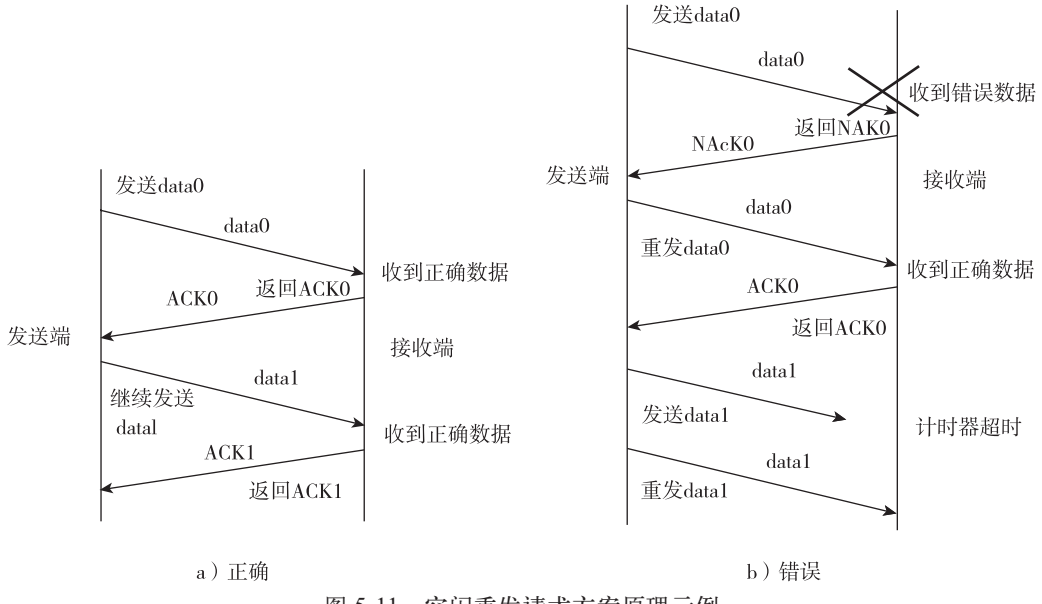


图 5-11　空闲重发请求方案原理示例

4）后续数据帧的发送步骤与 data0 帧的发送一样。

现在来打个与这里所介绍的空闲重发请求差错控制方案类似的比方。一个牙牙学语的小孩，你要他从 1 数到 10。由于他非常小，对这十个数字记得并不牢，所以他会一个个地去数，而且每数完一个数，他都会停下来，抬起头望着你，等待你的点头，或者说“对”，然后他才继续往下数。你对他点头或者说“对”，就相当于你向他发回了一个确认消息。

从以上过程可以看出，空闲重发请求方案除了要求在发送的数据帧中携带一定的检测错码外，还要求发送端和接收端都有一个用于存放待确认的发送帧和待向“网络层”提交的数据帧的缓冲存储空间，但这个存储空间比较小，因为它只需要临时存放一个数据帧。

由以上分析可以看出，这种 ARQ 方法设计简单，容易实现，但是这种方法也有一个不可克服的缺点，那就是每传送一个数据帧都要有一个等待时间（称为占空时间），信道的有效利用率低。占空时间与传送一个帧的全部时间的比例，称为占空比。数据帧越短，占空比就越大，相当于信道的利用率越低；数据帧越长，占空比就越小，信道的利用率就越高。但是数据帧越长，出错的概率也就越大，从而出现多次重发，因此传输效率也不会太高。正因如此才有下面将要介绍的连续重发请求差错控制方案。

5.3.5 连续重发请求方案

为了减小占空比，提高传输效率，人们又提出了连续重发请求（Continuous ARQ）的差错控制方案。

连续重发请求方案是指发送端可以连续发送一系列数据帧（也不总是不断地发送，具体可以连续发送多少个帧，要视双方的缓存空间大小，即窗口大小而定），即不用等前一帧被确认便可继续发送下一帧，效率大大提高。当然，在这个连续发送的过程中也可以接收来自接收端的响应消息（可以是确认帧，也可能是否认帧），发送端同样可以对传输出错的数据帧（如接收端返回了否认帧，或者响应计时器超时的帧）进行重发，具体处理方法后面会具体介绍。

由于连续重发请求方案减少了等待时间，整个通信的吞吐量就提高了，但在这种重发请求方案中，需要在发送端设置一个较大的缓冲存储空间（称为重发表），用以存放若干待确认的数据帧。当发送端得到某数据帧的确认帧后，便可从重发表中将该数据帧删除。

当出现传输差错时，连续重发请求方案有两种处理策略，即回退 N 帧（GO-DACK-N）策略和选择重发（Selective Repeat）策略。下面分别予以介绍。

1. 回退 N 帧策略

回退 N 帧策略的基本原理是，如果发送端一共发送了 n 个数据帧（编号从 0，一直到 $n-1$），但收到接收端发来的 ACK 确认帧中少了某一个或几个帧的 ACK 确认帧（要么是数据帧出现了丢失，要么是这些帧对应的 ACK 帧或者 NAK 帧出现了丢失，最终造成 ACK 确认帧序号不连续），或者在接收某一帧时检测出有错，接收端发送一个 NAK 否认帧给发送端；或者在计时器超时后仍没有收到某个帧的 ACK 或者 NAK 帧，则发送端可以判断接收端最后一个正确接收的帧编号，然后从缓存空间的重发表中重发所收到的最后一个 ACK 帧序

号以后的所有帧。

图 5-12 所示是一个回退 N 帧差错控制的示例。图中假定发送完 8 号帧后，0 号和 1 号帧的 ACK 帧已收到，但 2 号帧的 ACK 帧在计时器超时后还未收到，则发送端只能退回，从 2 号帧开始重发以后所有已发的数据帧（2~8 号共 7 个帧），尽管或许后面 3 ～ 8 号帧的 ACK 确认帧已收到。当然原来已发的这些帧接收端会自动删除的。

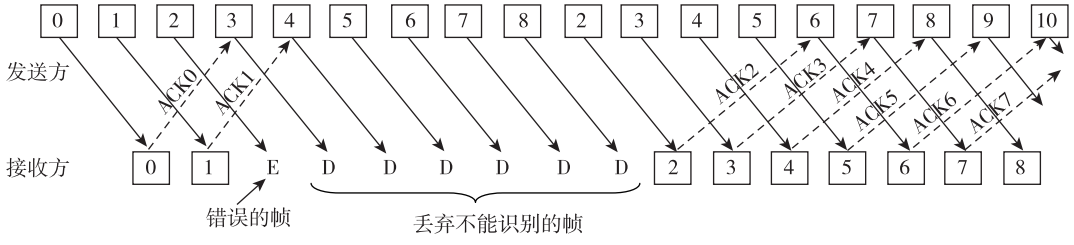


图 5-12　回退 N 帧策略差错控制原理示意图

下面我们将逐层深入分析“回退 N 帧”策略中的数据处理流程。

（1）理想情形下的数据处理流程

首先看一下在理想状态的情形，也就是数据帧和确认帧都不发生差错或丢失的情形（也就是不存在 NAK 帧返回，不存在数据帧和 ACK 帧丢失，也不存在计时器超时）下的数据处理流程：

1）发送端连续发送数据帧，而不等待任何数据帧的 ACK 帧的返回；

2）发送端在重发表中保存所发送的每个数据帧的备份；

3）接收端对每一个正确收到的数据帧返回一个 ACK 帧，ACK 帧中包括对应帧的编号；

4）接收端保存一个接收（次序）表，包含最后正确收到的数据帧的编号；

5）当发送端收到相应数据帧的 ACK 帧后，发送端即从重发表中删除该数据帧。

（2）存在帧差错情形下的数据流程

接下来考虑帧（包括数据帧和响应帧）出现差错的情形下回退 N 帧策略的数据处理流程，这里的“差错”既包括数据帧在接收端检测出的差错，也包括数据帧或响应帧在传输过程中出现丢失的差错。此时，回退 N 帧策略中的数据处理流程如下：

1）假设发送的第 N+1 个帧发生了差错，接收端要么检测出第 N+1 帧有错，要么发现没有接收到 N+1 帧，反而接收到了第 N+2 帧或第 N+3 帧，或后边其他帧；

2）出现这种情况时，接收端立即返回一个相应的未正确接收的否认帧 NAK（N+1），预示接收端最后正确收到的是第 N 帧（N+1 帧的前一帧）；同时对后面每个失序的数据帧，接收端都产生相应的 NAK 帧，否则如果所发送的 NAK（N+1）帧正好丢失或出错，将产生死锁，即发送端不停地发送新的帧，同时等待对第 N+1 帧的确认，而接收端不停地清除后继的帧，当然可以通过超时机制或者流量控制来避免死锁的发生（如滑动窗口法，具体将在本章后面介绍数据链路层的流量控制原理时介绍）；

3）发送端在收到 NAK（N+1）帧，或者收到了 NAK（N+2）、NAK（N+3）……帧时，从重发表中重发第 *N*+1 帧或者对应的 NAK 帧中序号所对应的帧，同时接收端清除所有失序的帧（从第 *N*+2 帧或者对应 NAK 帧中序号所对应的帧开始，直到重新正确接收到重发的第 *N*+1 或者对应 NAK 帧中序号所对应的帧）；

4）接收端重新收到第 *N*+1 帧，或者对应的 NAK 帧中序号所对应的帧，接收端就继续正常操作。

以上流程看似比较复杂，其实原理很简单。打个比方来说吧，比如我们经常要自己小孩从 1 数到 100，以检验他的数数能力。小孩正在数着，你突然发现某个数数得不正确（如数到 42 时本应是 43，却数成了 53），或者中间漏了一个或多个数（如数到 42 时，本应是 43 却数成跳过了，从 45 开始数了），你就会要求小孩从出现错误的那个数开始重数（如从 43 开始数），尽管在后面大多数是数得正确的。这就是回退 N 帧策略的差错控制原理。

2. 选择重发策略

回退 N 帧策略因可以连续发送数据帧而提高了传输效率，但也有不利的方面，那就是在重发时必须把原来已正确传送过的数据帧再次发送，仅仅是因为这些数据帧之前的某个数据帧或确认帧发生了差错，这样又使传输效率降低。所以当通信链路的传输质量很差、误码率较大时，回退 N 帧策略就没什么优势了，因为这时可能经常要重传大量的数据帧。

为了弥补回退 N 帧策略的不足，另一种效率更高的差错控制策略——选择重发策略诞生了。在这个差错控制策略中，当接收端发现某帧出错后，其后继续送来的正确帧虽然不能立即递交给接收端的“网络层”，但接收端仍可接收下来，先存放在一个缓冲区中，同时通过向发送端发送 NAK 否认帧，要求发送端重新传送出错的那一帧。一旦收到重新发来的正确帧后，就可以与原已存于缓冲区中的其余帧一起按正确的顺序递交给网络层。

选择重发策略规定，当发送端收到包含出错帧序号的 NAK 帧时，据此序号从重发表中选出相应帧的备份，直接插入到发送帧队列的前面给予重发，因为重发表的帧重发是按照 FIFO（先进先出）的机制进行排列的，插在前面是为了可以最先重发，避免了对后继正确数据帧的多余重发，使得传输效率明显提高了。

如果仍用小孩数数来打比方的话，就是你先让小孩自己一直数下去，发现数错时你记下来（不要打断他正常的数数流程），当小孩数完后你再根据这些错误要求小孩重数对应的数就行了，而不必要求他从错误的地方重新数下去。

下面也分两种情况来讨论“选择重发”策略的数据处理方法。

（1）数据帧出现差错情形下的处理流程

以下是当数据帧有差错（包括接收端检测到所接收的数据帧有差错，或者有数据帧丢失两种可能）时，“选择重发”策略的数据处理流程：

1）发送端连续发送多个数据帧，接收端对每个已正确接受的数据帧返回一个 ACK 帧；现假设第 N+1 个帧出现差错或丢失如果检测到第 N+1 个帧有错误，则向发送端返回一个否认 NAK（N+1）帧；如果一直到收到第 N+2 个数据帧时还没收到第 N+1 帧，则表明该帧已丢失，接收端不产生任何动作；但无论结果如何，已正确接收的数据帧，如第 N 个帧、第 N+2 个帧、第 N+3 个帧……仍会向发送方返回确认 ACK 帧；

2）当发送端收到来自接收端的否定 NAK（N+1）或者收到第 N+2 帧的 ACK 帧时，会检测出其失序（因为按顺序，在收到 ACK（N+2）帧之前应该是收到 ACK（N+1）帧），得知第 N+1 帧没有被确认。将第 N+2 帧从重发表中清除，并在继续发送后继数据帧之前重发第 N+1 帧。

图中 5-13 所示为由于接收端检测到 2 号帧有错，向发送端发送了一个否认帧 NAK2，要求发送端选择重发 2 号帧的示意图。从中可以看限出，“选择重发”策略下只需发送有错的帧，而不会发送从有错帧开始后面所的帧，显然减少了信道资源浪费，提高了传输效率，但要求发送端和接收端都有足够大的缓冲区空间，以便存储多个帧的重发表和预提交数据帧。

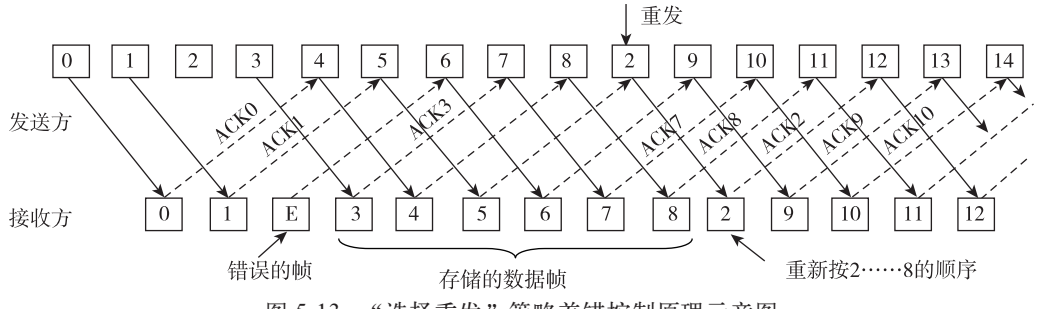


图 5-13　“选择重发”策略差错控制原理示意图

（2）响应帧出现差错情形下的数据处理流程

在响应帧（包括确认 ACK 帧和否认 NAK 帧）出现差错时，也就是本应接收的是第 N 个帧的响应帧却接收到了第 N+1 个帧的响应帧情况下，在该情形下数据处理流程如下（现以 ACK 帧出错进行介绍，与 NAK 帧出错的处理流程一样）：

1）当发送端已到了第 $N-1$ 个帧的 ACK 帧，接下来应该收到的是第 N 个帧的 ACK 帧，而偏偏收到的是第 N+1 个帧的 ACK 帧；

2）发送端在收到 ACK（N+1）帧后，检测出在重发表中第 N 个帧都还没收到 ACK 帧，因此认为第 N 个帧出现了差错（事实上并不是这样的），重发第 N 个帧；

3）接收端在收到发送端重发的第 N 个帧，搜索接收表并确定第 N 帧已被正确接收，因此认定这个重发的第 N 帧是重复的，于是直接删除这个重发的第 N 帧，并返回一个 ACK（N）给发送端，以使发送端从重发表中删除第 N 帧。这样就达到了响应帧出现差错时的错误纠正。

5.3.6 海明纠错码

海明码（Hamming Code）是一个可以有多个校验位，具有检测并纠正一位错误代码功能的纠错码，所以它也仅用于信道特性比较好的环境中，如以太局域网中，因为如果信道特性不好的情况下，出现的错误通常不是一位。

海明码的检错、纠错基本思想是将有效信息按某种规律分成若干组，每组安排一个校验位进行奇偶性测试，然后产生多位检测信息，并从中得出具体的出错位置，最后通过对错误位取反（也是原来是 1 就变成 0，原来是 0 就变成 1）来将其纠正。

要采用海明码纠错，需要按以下步骤来进行：计算校验位数→确定校验码位置→确定校验码→实现校验和纠错。下面来具体介绍这几个步骤。

1. 计算校验位数

要使用海明码纠错，首先就要确定发送的数据所需要的校验码（也就是“海明码”）位数（也称校验码长度）。它是这样的规定的：假设用 N 表示添加了校验码位后整个信息的二进制位数，用 K 表示其中有效信息位数，r 表示添加的校验码位，它们之间的关系应满足：$N=K+r\leqslant 2^r-1$。

如 $K=5$，则要求 $2^r-r\geqslant 5+1=6$，根据计算可以得知 r 的最小值为 4，也就是要校验 5 位信息码，则要插入 4 位校验码。如果有效信息位数是 8，则要求 $2^r-r\geqslant 8+1=9$，根据计算可以得知 r 的最小值也为 4。根据经验总结，得出信息码和校验码位数之间的关系如表 5-1 所示。

表 5-1　信息码位数与校验码位数之间的关系

信息码位数	1	2 ～ 4	5 ～ 11	12 ～ 26	27 ～ 57	58 ～ 120	121 ～ 247
校验码位数	2	3	4	5	6	7	8

2. 确定校验码位置

上一步我们确定了对应信息中要插入的校验码位数，但这还不够，因为这些校验码不是直接附加在信息码的前面、后面或中间的，而是分开插入到不同的位置的。但不用担心，校验码的位置很容易确定的，那就是校验码**必须**是在 2^n 次方位置，如第 1 位，2 位，4 位，8 位，16 位，32 位……（对应 2^0，2^1，2^2，2^3，2^4，2^5，……，从最左边的位数起），这样一来就知道了信息码的分布位置，也就是非 2^n 次方的位置，如第 3 位，5 位，6 位，7 位，9 位，10 位，11 位，12 位，13 位，……（从最左边的位数起）。

举一个例子，假设现有一个 8 位信息码，即 b1、b2、b3、b4、b5、b6、b7、b8，由表 5-1 得知，它需要插入 4 位校验码，即 p1、p2、p3、p4，也就是整个经过编码后的数据码（称为码字）共有 12 位。根据以上介绍的校验码位置分布规则可以得出，这 12 位编码后的数据就是 p1、p2、b1、p3、b2、b3、b4、p4、b5、b6、b7、b8。

假设原来的 8 位信息码为 10011101，因还没有求出各位校验码值，现在这些校验码位都用“？”表示，最终的码字为：？？1？001？1101。

3. 确定校验码

经过前面的两步，我们已经确定了所需的校验码位数和这些校验码的插入位置，但这还不够，还得确定各个校验码值。这些校验码的值不是随意的，每个校验位的值代表了代码字中部分数据位的奇偶性（最终要根据是采用奇校验还是偶校验来确定），其所在位置决定了要校验的比特位序列。总的原则是：第 i 位校验码从当前位开始，每次连续校验 i（这里是数值 i，不是第 i 位，下同）位后再跳过 i 位，然后再连续校验 i 位，再跳过 i 位，以此类推。最后根据所采用的是奇校验还是偶校验即可得出第 i 位校验码的值。

（1）计算方法

校验码的具体计算方法如下：

p1（第 1 个校验位，也是整个码字的第 1 位）的校验规则是：从当前位数起，校验 1 位，然后跳过 1 位，再校验 1 位，再跳过 1 位，……。这样就可得出 p1 校验码位可以校验的码字位包括：第 1 位（也就是 p1 本身），第 3 位，第 5 位，第 7 位，第 9 位，第 11 位，第 13 位，第 15 位，……。然后根据所采用的是奇校验还是偶校验，最终可以确定该校验位的值。

p2（第 2 个校验位，也是整个码字的第 2 位）的校验规则是：从当前位数起，连续校验 2 位，然后跳过 2 位，再连续校验 2 位，再跳过 2 位，……。这样就可得出 p2 校验码位可以校验的码字位包括：第 2 位（也就是 p2 本身），第 3 位，第 6 位，第 7 位，第 10 位，第 11 位，第 14 位，第 15 位，……。同样根据所采用的是奇校验还是偶校验，最终可以确定该校验位的值。

p3（第 3 个校验位，也是整个码字的第 4 位）的校验规则是：从当前位数起，连续校验 4 位，然后跳过 4 位，再连续校验 4 位，再跳过 4 位，……。这样就可得出 p4 校验码位可以校验的码字位包括：第 4 位（也就是 p4 本身），第 5 位，第 6 位，第 7 位，第 12 位，第 13 位，第 14 位，第 15 位，第 20 位，第 21 位，第 22 位，第 23 位，……。同样根据所采用的是奇校验还是偶校验，最终可以确定该校验位的值。

p4（第 4 个校验位，也是整个码字的第 8 位）的校验规则是：从当前位数起，连续校验 8 位，然后跳过 8 位，再连续校验 8 位，再跳过 8 位，……。这样就可得出 p4 校验码位可以校验的码字位包括：第 8 位（也就是 p4 本身），第 9 位，第 10 位，第 11 位，第 12 位，第 13 位，第 14 位，第 15 位，第 24 位，第 25 位，第 26 位，第 27 位，第 28 位，第 29 位，第 30 位，第 31 位，……。同样根据所采用的是奇校验，还是偶校验，最终可以确定该校验位的值。

……

我们把以上这些校验码所校验的位分成对应的组，它们在接收端的校验结果（通过对各校验位进行逻辑“异或运算”得出）对应表示为 G1、G2、G3、G4，.....（正常情况下均为 0）。

（2）校验码计算示例

同样举上面的例子来说明，码字为？？1？001？1101。

先求第 1 个“？”（也就是 p1，第 1 位）的值，因为整个码字长度为 12（包括信息码长和校验码长），所以可以得出本示例中 p1 校验码校验的位数是 1、3、5、7、9、11 共 6 位。这 6 位中除了第 1 位（也就是 p1 位）不能确定外，其余 5 位的值都是已知的，分别为 1、0、1、1、0。现假设采用的是偶校验（也就是要求整个被校验的位中的“1”的个数为偶数），从已知的 5 位码值可知，已有 3 个“1”，所以此时 p1 位校验码的值必须为“1”，得出 p1=1。

再求第 2 个“？”（也就是 p2，第 2 位）的值，根据以上规则可以很快得出本示例中 p2 校验码校验的位数是 2、3、6、7、10、11，也是一共 6 位。这 6 位中除了第 2 位（也就是 p2 位）不能确定外，其余 5 位的值都是已知的，分别为 1、0、1、1、0。现假设采用的是偶校验，从已知的 5 位码值可知，也已有 3 个“1”，所以此时 p2 位校验码的值必须为“1”，得出 p2=1。

再求第 3 个“？”（也就是 p3，第 4 位）的值，根据以上规则可以很快得出本示例中 p3 校验码校验的位数是 4、5、6、7、12，一共 5 位。这 5 位中除了第 4 位（也就是 p3 位）不能确定外，其余 4 位的值都是已知的，分别为 0、0、1、1。现假设采用的是偶校验，从已知的 4 位码值可知，也已有 2 个“1”，所以此时 p2 位校验码的值必须为“0”，得出 p3=0。

最后求第 4 个“？”（也就是 p4，第 8 位）的值，根据以上规则可以很快得出本示例中 p4 校验码校验的位数是 8、9、10、11、12（本来是可以连续校验 8 位的，但本示例的码字后面的长度没有这么多位，所以只校验到第 12 位止），也是一共 5 位。这 5 位中除了第 8 位（也就是 p4 位）不能确定外，其余 4 位的值都是已知的，分别为 1、1、0、1。现假设采用的是偶校验，从已知的 4 位码值可知，已有 3 个“1”，所以此时 p2 位校验码的值必须为“1”，得出 p4=1。

最后就可以得出整个码字的各个二进制值码字为 111000111101（带阴影的 4 位就是校验码）。

4. 实现校验和纠错

虽然上一步已把各位校验码求出来了，但是如何实现检测出哪一位在传输过程中出了差错呢？（海明码也只能检测并纠正一位错误）它又是如何实现对错误的位进行纠正的呢？其实最关键的原因就是海明码是一个多重校验码，也就是码字中的信息码位可同时被多个校验码校验，然后通过这些码位对不同校验码的联动影响最终可以找出是哪一位出错了。

（1）海明码的差错检测

现假设整个码字一共是 18 位，根据表 5-1 可以很快得出，其中有 5 位是校验码，再根据本节前面介绍的校验码校验规则可以很快得出各校验码所校验的码字位，如表 5-2 所示。

表 5-2　各校验码校验的码字位对照表

码字中的位	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
对应的位	p1	p2	b1	p3	b2	b3	b4	p4	b5	b6	b7	b8	b9	b10	b11	p5	b12	b13
p1 校验的位	√		√		√		√		√		√		√		√		√	
p2 校验的位		√	√			√	√			√	√			√	√			√
p3 校验的位				√	√	√	√					√	√	√	√			
p4 校验的位								√	√	√	√	√	√	√	√			
p5 校验的位																√	√	√

注：不带阴影的勾位是对应的校验码所在的位，而带阴影的勾位是该校验码要进行校验的数据位。

从表中可以得出以下两个规律：

- 所有校验码所在的位是由对应的校验码进行校验的，如第 1 位（只由 p1 校验）、第 2 位（只由 p2 校验）、第 4 位（只由 p3 校验）、第 8 位（只由 p4 校验）、第 16 位（只由 p5 校验），……也就是这些位如果发生了差错，影响的只是对应的校验码的校验结果，不会影响其他校验码的校验结果。**这点很重要，如果最终发现只是一个校验组中的校验结果不符，则直接可以知道是对应校验组中的校验码在传输过程中出现了差错**。
- 所有信息码位均被至少两个校验码校验了，也就是至少校验了两次。查看对应的是哪两组校验结果不符，然后根据表 5-2 就可以很快确定是哪位信息码在传输过程中出了差错。

海明码校验的方式就是各校验码对它所校验的位组进行异或运算，即：

G1=p1 ⊕ b1 ⊕ b2 ⊕ b4 ⊕ b5 ⊕……

G2=p2 ⊕ b1 ⊕ b3 ⊕ b4 ⊕ b6 ⊕ b7 ⊕ b10 ⊕ b11 ⊕……

G3= p3 ⊕ b2 ⊕ b3 ⊕ b4 ⊕ b8 ⊕ b9 ⊕ b10 ⊕ b11 ⊕……

G4= p4 ⊕ b5 ⊕ b6 ⊕ b7 ⊕ b8 ⊕ b9 ⊕ b10 ⊕ b11 ⊕……

G5= p5 ⊕ b12 ⊕ b13 ⊕ b14 ⊕ b15 ⊕ b16 ⊕ b17 ⊕ b18 ⊕ b19 ⊕ b20 ⊕ b21 ⊕ b11 ⊕ b23 ⊕ b24 ⊕ b25 ⊕ b26 ⊕……

正常情况下（也就是整个码字不发生差错的情况下），在**采用偶校验时**，各校验组通过异或运算后的**校验结果均应该是为 0**，也就是前面所说的 G1、G2、G3、G4，……均为 0，因为此时 1 为偶数个，进行异或运算后就是 0；而**采用奇校验时，各组校验结果均应是为 1**。

现在举一个例子来说明，假设传输的海明码为 111000111101（一共 12 位，带阴影的 4 位就是校验码），从中可以知道它有 4 个校验组：G1、G2、G3、G4，然而到达接收端经过校验后发现只有 G4=1（也就是只有这组校验结果不等于 0），通过前面介绍的校验规律可以很快地发现是 G4 校验组中的 p4 校位码（也就是整个码字中的第 8 位）错了（因为只有一组校验结果出现差错时，则肯定只是对应的校验位出了差错），也就是最终的码字变成了：

111000001101。

再假设 G3、G4 两个校验值都不为 0，也就是都等于 1。通过表 5-2 所示比较 G3、G4 两个校验组（注意本示例中码字长度一共才 12 位，只需要比较前 12 位）中共同校验的码位可以很快发现是 b8，也就是第 12 位出现了差错，也就是最终的码字变成了：111000011100。

经验之谈 这里一定要注意，最终有多少个校验组出现差错也不是随意的，一定要结合实际传输的码字长度来考虑。如上例一共 12 位，如果换成了 16 位的码字，且当 b9 位出现差错时，则 G1、G3、G4 一定会同时出现错误，因为 b9 这个位是三个校验组同时校验的，只要它一出错，肯定会同时影响这三个校验组的值。同理，如果是 b11 位出现了差错，因为它同时受 G1、G2、G3、G4 四个校验组校验，所以这四个校验组结果都将出现错误。

（2）海明码的差错纠正

检测出是哪位出现了差错还不够，因为海明码具有纠正一位错误的能力，所以还需要完成纠错过程。这个过程的原理比较简单，就是直接对错误的位进行取反或者加“1”操作，使它的值由原来的“1”变成“0”，或由原来的“0”变成“1”（因为二进制中每一位只能是这二者之一）。

以上就是海明码的差错检测和差错纠正原理了，虽然比单纯的奇偶校验码复杂些，但只要理清了思路，还是比较简单的。

5.4 流量控制

介绍完复杂的数据链路层“差错控制”功能后，接下来介绍数据链路层的“流量控制”功能。其实在上节介绍差错控制功能时就提到了流量控制功能，因为一些差错控制方案本身就具有一定的流量控制功能，如前面介绍的空闲重发请求方案中规定，发送端每发完一个数据帧后把这个帧保存在缓存空间中，然后就停下来等待接收端发来的确认消息，然后才能继续发送下一帧，这就可以控制路中的数据流量。在连续重发请求方案中，虽然发送端可以一次发送多个数据帧，但是也不是没有限制的，因为发送端需要把每次发送的数据帧保存在缓存空间中，接收端也要把向网络层提交的数据帧先保存在缓存空间中，所以最终一次能发送多少个帧，是由双方缓存空间大小决定的。这就是本节后面将要提到的“窗口大小”。

数据链路层的流量控制方案主要有两种：一种是适用于面向字符的异步通信协议（如 RS—232）中的简单流量控制方案——XON/XOFF（继续 / 停止）方案；另一种是适用于大量数据通信环境中的“滑动窗口机制”。

5.4.1 XON/XOFF 流量控制方案

XON/XOFF（transmitter on/transmitter off，继续传输 / 停止传输）是一种流量控制协议，常用于数据传输速率大于等于 1200bps，而接收端数据处理速率远小于这个值（也就是通信

双方速率不同步）的情形，通过对发送端的数据传输速率进行控制，以达到与接收数据数据处理速率匹配。

XON/XOFF（继续 / 停止）是一种最简单的流量控制技术，主要适用于异步通信中，接收端通过使用特殊字符来控制发送端数据的发送。其基本思想是：当接收端认为不能继续接收数据时（也就是接收端的缓存空间满了或者接近满时），接收端会向发送端发送一个 XOFF 控制字符，当发送端收到对应的 XOFF 控制字符时就停止数据的继续发送；当接收端可以继续接收数据时，接收端会再向发送端发送一个 XON 控制字符，发送端收到这个控制字符后就知道可以恢复数据发送了，继续发送数据，一直这么循环下去。

其中 XON 采用 ASCII 字符集中的控制字符 DC1（十进制值为 17，十六进制值为 11，相当于按下“Ctrl+Q”组合键），XOFF 采用 ASCII 字符集中的控制字符 DC3（十进制值为 19，十六进制值为 13，相当于按下“Ctrl+S”组合键）。在一次数据传输过程中，XOFF、XON 的周期可重复多次，但这些操作对用户来说是透明的，也就是说用户不用管它，设备会自动操作。

许多异步数据通信软件均支持 XON/XOFF 协议。这种方案也可用于计算机向打印机或其他终端设备（如 Modem 的串行通信）发送字符，在这种情况下，打印机或终端设备中的控制部件用以控制字符流量。如我们在通过 Modem 拨号连接网络时，采用的就是这种流量控制方法。当从 PC 机上的数据到达 Modem 时，如果 Modem 中的缓存空间满了，它就会向 PC 机发送一个代表“停止传输”的 XOFF 控制字符，而当 Modem 中的缓存空间没满时，Modem 又会向 PC 机传送一个代表“继续传输”的 XON 控制字符。

再如，在局域网中一台 PC 机连接了一台打印速度比较慢的打印机，当 PC 机开始向打印机发送要打印的文件时，因为 PC 机的数据传输速率非常高，有许多文件打印机很难及时打印。此时打印机会向 PC 机发送一个 XOFF 字符来通知 PC 机，要求暂停文件的发送。PC 机上的软件看到来自打印机的 XOFF 控制字符后，就会暂停文件的发送；而当打印机中排队等候打印的文件打完了，或者打印得差不多了时，打印机又会向 PC 机发送一个 XON 控制字符，通知 PC 机可以继续发送要打印的文件。

5.4.2　滑动窗口机制

在上面介绍的 XON/XOFF 流量控制方案中，为了实现发送端与接收端的速率匹配，需要往返发送一些特殊的控制字符，这样就会使得信道的利用率大打折扣，其主要是用于与一些低数据处理能力的接收端通信时采用。在实际的数据链路层流量控制中，更多的是采用本节将要介绍的“滑动窗口机制”来进行流量控制的（传输层也有这样的流量控制方案，具体将在第 10 章介绍）。

1. 理解“滑动窗口”机制

“滑动窗口机制”中的“窗口”是指发送端和接收端的缓存空间大小；“滑动”的意思是指缓存空间中存放的未处理帧数是变化的，发送端在收到确认帧后会删除原来保存在缓存中

的待重发帧，而接收端向网络层提交一个帧后也会删除原来保存在缓存中的帧。

至于缓存空间大小，采取不同的流量控制方案其会有不同的值，但要明白的是，缓存空间都是非常有限的，就像计算机 CPU 中的缓存一样。缓存越大，成本越高。如在 5.3.4 节介绍的空闲重发请求方案中，一次只能发送一个帧，发完一个帧后就等待来自接收端的确认帧，收到确认帧后就删除原来保存在缓存空间中的待重发帧，接收端不需要缓存空间，因为它接收到数据后即进行处理，对于有错误的帧直接丢弃。所以在“空闲重发请求”方案中仅发送端需要保存一个帧的缓存空间，也就是它的“缓存空间大小”就是一个帧。

而在前面介绍的“连续重发请求”方案中，发送端一次可以连续发送多个帧，并且在其缓存空间中保存所有已发，但没有接收到来自接收端确认帧的待重发帧，而不用像“空闲重发请求”那样发一帧停下来等待接收端的确认帧；接收端也可以在缓存空间中保存来不及处理的帧，大大提高了数据传输的效率。

同样打个简单的比喻来说。就像一个水缸接了大小两个不同口径的水管，进水管的口径较大，出水管的口径较小。在这样的情况下，进水管肯定不能持续不断地向水缸中加水，否则就会因出水速率不匹配导致水从水缸中流出来了。所以通常是进水管供了一段时间的水后就要停下来，等水缸中的水用得差不多了再加水。这时水缸的容量就相当于上面所说的接收端缓存空间大小了。

这里涉及一个非常重要的问题，那就是到底一次最多连续发多少个帧比较合适呢？这里要考虑两方面的因素，一是要能实现有效的差错控制，二是要与接收端的数据处理能力相匹配，前者我们已介绍，本节仅从后者来考虑，也就是从流量控制角度来考虑。缓存空间大小又与帧编号有关，因为在数据传输时，每个帧都是有序列号的，这在本章前面介绍“差错控制”方案中就已说到。缓存空间越大，用于帧编号的位数就要越多，如 1 位可以表示 2 个帧（也就是窗口大小为 2），2 位可以表示 4 个帧（也就是窗口大小为 4），3 位可以表示 8 个帧（也就是窗口大小为 8），……。但用于指示帧序列号的位数越多，帧的无用开销就越大，所以在一般的数据链路层协议中只有 2 位用于帧编号。

2. 滑动窗口实例

在此以 1 位帧序列号，也就是窗口大小为 2 的示例来介绍“滑动窗口机制”。下面从初始状态开始介绍整个流程（注意，这里仅考虑正常传输情况，不考虑出现差错的情况），如图 5-14 所示（各步对应图中的相应序号）：

1）在初始状态下，发送端和接收端的缓存空间中均没有保存数据帧。

2）发送端开始发送第一个帧——0 号帧，并把它保存在缓存空间中，并建立一个待重发表，第一个帧号就是“0”。

3）因为是 1 个比特位用于帧编号，窗口大小为 2，所以发送端还可以继续发送第二个帧——1 号帧。同时把这个帧保存在缓存空间中，并向待重发表中添加第二个帧的序号“1”。此时因为在发送端的缓存空间中已保存了两个帧（0 号帧和 1 号帧），达到了窗口大小的值，不能继续发送后面的帧了，先停下来等待来自接收端的确认帧。在等待的过程中，接收端可

能收到了 0 号帧，然后向发送端返回 0 号帧的确认消息。

4）发送端在收到 0 号帧的确认帧后，立即从缓存空间中清除原来保存的 0 号待重发帧，并在待重发表中清除帧序号“0”；接收端把收到的 0 号帧提交给网络层，并清除缓存空间中的 0 号帧，此时缓存空间中又为空了。

5）此时发送端继续发送第三个帧——2 号帧，同时也把这个帧保存在缓存空间中，并向待重发表中添加第三个帧的序号“2”。因为此时发送端的缓存空间中又已保存了两个帧（1 号帧和 2 号帧），又达到了窗口大小的值，不能继续发送后面的帧了，要再先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能又收到了 1 号帧，然后向发送端返回 1 号帧的确认帧。

6）发送端在收到 1 号帧的确认帧后，立即从缓存空间中清除原来保存的 1 号待重发帧，并在待重发表中清除帧序号“1”。接收端把收到的 1 号帧提交给网络层，清除缓存空间中的 1 号帧，此时缓存空间中又为空了。

7）此时发送端继续发送第四个帧——3 号帧，同时也把这个帧保存在缓存空间中，并向待重发表中添加第四个帧的序号“3”。同样，因为此时发送端的缓存空间中又已保存了两个帧（1 号帧和 2 号帧），又达到了窗口大小的值，不能继续发送后面的帧了，再要先停下来等待来自接收端的确认帧。在等待的过程中，接收端可能又收到了 2 号帧，然后向发送返回 2 号帧的确认帧。

8）发送端在收到 2 号帧的确认帧后，立即从缓存空间中清除原来保存的 2 号待重发帧，并在待重发表中清除帧序号“2”。接收端把收到的 2 号帧提交给网络层，清除缓存空间中的 2 号帧，此时缓存空间中又为空了；

……

后面的步骤按照以上流程类推。

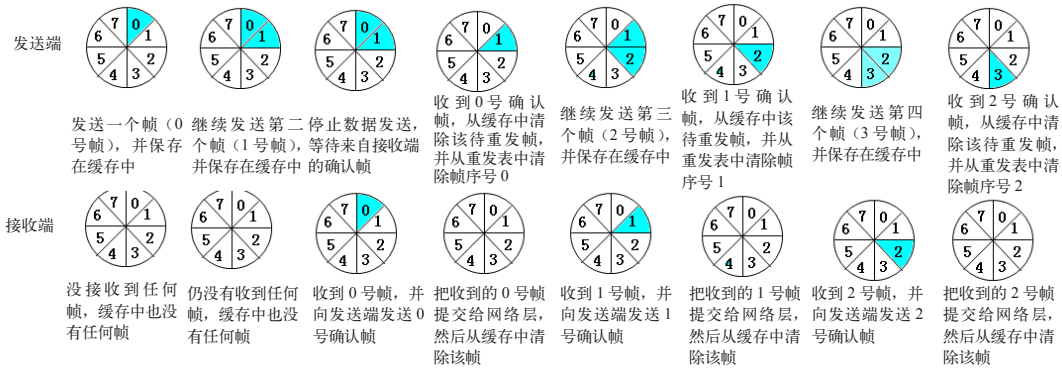


图 5-14　“滑动窗口机制”工作流程示例

说明　数据链路层中常见的协议可分为两大类：面向字符的链路层协议和面向比特

的链路层协议（这些都属于同步传输模式协议）。面向字符的链路层协议主要有 IBM BSC（Binary Synchronous Communication，二进制同步通信）协议、DEC DDCMP（Digital Data Communications Message Protocol，数字数据通信消息协议）、SLIP（Serial Line Internet Protocol，串行线路网际协议）、PPP（Point-to-Point Protocol，点对点协议）等；面向比特的链路层协议主要有 IBM 的 SDLC、ANSI 通过修改 SDLC 而提出的 ADCCP（Advanced Data Communication Control Procedure，高级数据通信控制规程）、ISO 通过修改 SDLC 而提出 HDLC（High-level Data Link Control，高级数据链路控制）、CCITT 通过修改 HDLC 而提出 LAP（Link Access Procedure，链路访问规程）等。下面将介绍几种典型的数据链路层协议。

5.5 面向字符的 BSC 协议

面向字符的同步方法也称“字符填充的首尾定界符法”。在该同步方法中，数据帧中的数据都被看作字符序列（所以称之为面向字符的同步传输），所有的控制信息也都是字符形式（当然数据的表示形式还是二进制的比特流），每个数据块的头部用一个或多个同步字符 SYN 来标记数据块的开始；尾部用字符 ETX 来标记数据决的结束。

面向字符的同步传输协议的典型代表就是 IBM 公司的 BSC 协议。BSC 协议规定，链路上传送的数据必须是由规定字符集（可以是 ASCII，或者 EBCDIC（Extended Binary Coded Decimal Interchange Code，扩展二进制－十进制交换码））中的字符组成，控制信息也必须由同一个字符集中的若干指定的控制字符构成。

5.5.1 BSC 控制字符和数据块结构

BSC 协议与所有同步传输协议一样，也是一次可以传送由若干个字符组成的数据块（通常是一帧），而不是一次只传送一个字符。同时规定了十种特殊字符（称为通信控制字符）作为这个数据块的开始与结束标志，以及整个传输过程的各种控制信息标志（并不是每个数据块中都有这十种全部的控制字符）。这十种通信控制字符说明如下：

- ACK（Acknowledge）：确认标志，由接收端发出的，作为对正确接收到报文的响应。
- DLE（Data Link Escape）：转义标志，用于指示后面的字符是数据字符，而不是特殊控制字符。这是用来进行透明传输的，当在报文中也存在这十个控制字符时，在这些字符前加上 DLE 字符后，通知接收端把它们当作普通的报文处理，而不是作为控制字符来识别。具体将在本节后面介绍。
- ENQ（Enquire）：询问标志，用于请求远程站点给出响应。响应可能包括远程站点的身份或状态。
- EOT（End of Transmission）：发送完毕标志，用于表示一个或多个文本的发送结束，并拆除链路。
- ETB（End of transmission Block）：块终止或组终止标志，用于标志每个数据块的结

束位置。仅在一个报文要分成多个数据块传输时才有此标志。

- ETX（End of Text）：文本终止标志，标志报文文本的结束。仅在一个报文不分成多个数据块传输时才有此标志。
- NAK（Negative Acknowledge）：否认标志，由接收端发出的，作为对未正确接收的报文响应。
- SOH（Start of Head）：报头开始标志，用于表示报文的标题信息或报头的开始。仅在报文的第一个数据块中才有此标志。
- STX（Start of Test）：文本开始标志，标志标题信息的结束和报文文本的开始。每个数据块均有此标志。
- SYN（Synchronous）：字符同步标志，用以实现通信双方的字符同步，或用于在无数据传输时保持同步。在每个数据块中均有此标志，而且通常是两个。

以上这十种通信控制字符所对应的 ASCII 码（ASCII 中是用低 7 位表示一个字符的，最高位为校验码）或 EBCDIC 码值如表 5-3 所示。这些控制字符代码所对应的 ASCII 也可参见图 4-7。这种通信控制字符中，在数据同步传输中起关键作用的就是 SYN、SOH、STX、ETB、ETX、EOT 这六种通信控制字符。

表 5-3　BSC 协议十种通信控制字符对应的代码

控制字符名称	对应的 ASCII 码	对应的 EBCDIC 码
ACK	0000110	00101110
DLE	0010000	00010000
ENQ	0000101	00101101
EOT	0000100	00110111
ETB	0010111	00100110
ETX	0000011	00000011
NAK	0010101	00111101
SOH	0000001	00000001
STX	0000010	00000010
SYN	0010110	00110010

通过前面的介绍可以知道，根据同步传输中报文在数据块中所处的位置不同，BSC 协议的数据块有所不同，具体有如下四种格式。

- 不带报头的单块报文或分块传输中的最后一块报文的格式为：

SYN	SYN	STX	报文	ETX	BCC

- 带报头的单块报文的格式为：

SYN	SYN	SOH	报头	STX	报文	ETX	BCC

❑ 分块传输中的第一块报文的格式为：

SYN	SYN	SOH	报头	STX	报文	ETB	BCC

❑ 分块传输中的中间报文的格式为：

SYN	SYN	STX	报文	ETB	BCC

从以上四种数据报文格式中可以得出 BSC 报文传输的如下两个基本特点：① BSC 协议中所有发送的数据均跟在至少两个 SYN 字符之后，以使接收端能实现字符同步；②所有数据块在块终止标志符（ETX 或 ETB）之后还有块校验字符 BCC（Block Check Character），校验是单字节的 CRC（循环校验码）或双字节的 CRC，校验范围从 STX 开始到 ETX 或 ETB 为止。

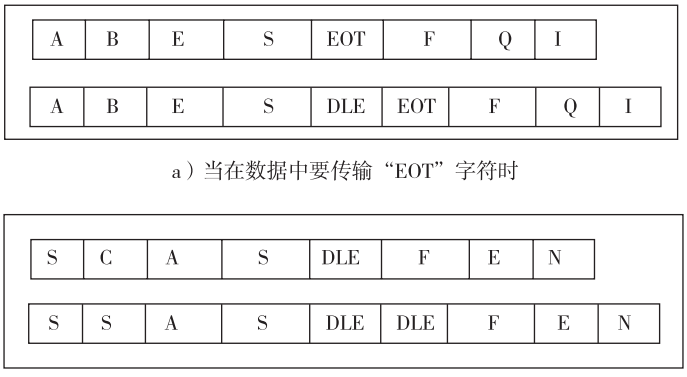
5.5.2 BSC 协议数据透明传输原理

面向字符的同步传输协议不像异步传输协议那样需要在每个字符前后附加起始和停止位，因此传输效率提高了。但由于采用了一些特殊的传输控制字符，在增强了通信控制能力和校验功能的同时也带来了新的问题，那就是如何区别数据字符代码和特殊控制字符代码。因为在数据块中完全有可能出现与特殊控制字符代码相同的数据字符，这就会在接收端产生误解。比如正文有个与终止字符 ETX 的代码相同（在 ASCII 码中为 0000011）的数据字符，如果不做任何处理，接收端就不会把它当作普通数据处理，而误认为是正文结束，进而产生差错。

为此解决这一问题，面向字符的同步传输协议应具有将数据块中的特殊字符作为普通数据处理的能力，这种能力称为数据透明。解决方案是在同步传输协议中设置转义字符 DLE（Data Link Escape，数据链路封装）。当需要在数据块中传输一个与某个特殊字符代码一样的数据时，就要先在它前面要加一个 DLE 代码（ASCII 码中为 0010000），这样接收端在收到一个 DLE 代码时，就知道了它下面一个字符是普通的数据字符，而不是通信控制字符。在接收端如果发现一个与某个特殊字符代码相同的字符是 DLE 代码，则直接删除前面的 DLE 代码，仅接收真实的数据部分。

如要在数据中传输 EOT 字符，为了避免把它误认为是通信控制字符，就是需先键入一个 DLE 字符，然后再键入 EOT 字符，如图 5-15a 所示。在接收端会直接删除 EOT 字符代码前面的 DLE 代码，仅接收数据部分的 EOT 代码。

同样，当要传输的数据中有一个代码与 DLE 的数据字符相同时，也要先在它前面加上另外一个 DLE 字符，相当于要传输一个普通的 DLE 数据字符，即需要键入两个“DLE”字符，如图 5-15b 所示。在接收端如果发现有连续两个 DLE 字符，则直接删除前面那个 DLE 字符，仅接收后面一个作为数据的 DLE 字符。



a）当在数据中要传输“EOT”字符时

b）当在数据中要传输“DLE”字符时

图 5-15　两个数据透明传输示例

从以上可以看出，这种面向字符的同步方法实现起来相当麻烦，因为 BSC 这类同步传输协议中的控制字符本身比较多，很容易就造成了与数据中传输数据代码的冲突。也正是因为这个缺点，后面又产生了下节将要介绍的面向比特的同步传输协议。

5.6　面向比特的 SDLC 和 HDLC 协议

面向比特的同步传输协议中，数据块是作为比特流来处理的（而不是作为字符流来处理），所以称之为面向比特的同步传输。在面向比特的同步传输中，每个数据块的头部和尾部都用一个特殊的比特序列（如 01111110）来标记数据块的开始和结束，这就是面向比特的同步传输协议中的基本成帧原理，或者说是基本的帧同步原理。在局域网中所采用的传输方式都是面向位流的同步传输方式，由它们各自的介质访问控制协议来定义具体的数据格式（即帧格式）以及相应的介质访问控制方法。

面向比特同步传输的通信协议中，最具有代表性的是 IBM 的 SDLC（Synchronous Data Link Control，同步数据链路控制）协议、国际标准化组织 ISO 的 HDLC 协议，美国国家标准协会 ANSI 的 ADCCP。

SDLC 协议是一种 IBM 数据链路层协议，适用于系统网络体系结构（SNA）；HDLC 是一种在同步网上传输数据、面向位的数据链路层协议，是 ISO 对 IBM 的 SDLC 协议进行了改进和标准化的协议。但是 SDLC 属于单链路规程，而 HDLC 属于多链路规程。所有面向比特的数据链路控制（DLC）协议均采用统一的帧格式，且不论是数据还是单独的控制信息均以帧为单位传送。

SDLC 支持识别两类网络站点：主站点（Primary Station）和从站点（Secondary Station）。主站点控制从站点，主站点轮询从站点是否有数据要发送。也就是说，从站点只有在主站点授权前提下才可以向主站点发送信息，如果一个从站点有数据要发送，当它被主站点识别后才可开始数据传输。主站点按照预先确定的顺序选择从站点，一旦选定的从站点已经导入数据，

那么它即可进行数传输。同时主站点可以建立和拆除链路，并在运行过程中控制这些链路。

5.6.1 HDLC 链路结构和操作方式

HDLC 是也采用主站点和从站点操作方式。在链路上起控制作用的站点称为主站点，其他的受主站控制的站点称为从站点。主站点负责对数据流进行组织，并且对链路上的差错实施恢复。由主站点发往从站点的帧称为命令帧，而由从站点返回主站点的帧称为响应帧。连有多个站点的链路通常使用查询技术，对其他站点进行查询的站点称为主站点，而在点到点链路中每个站点均可作为主站点。主站需要比从站点有更多的功能，所以当终端与主机相连时，主机一般总是主站点；在一个站点连接多条链路的情况下，该站点对于一些链路而言可能是主站点，而对另外一些链路而言又可能是从站点。

在 HDLC 中，对主站点、从站点和复合站点定义了图 5-16 所示的三种链路结构：不平衡链路结构、对称非平衡链路结构和平衡链路结构。从图中可以看出，不同链路结构，允许的站点的类型以及各站点发送命令帧和响应帧的权限也不一样。

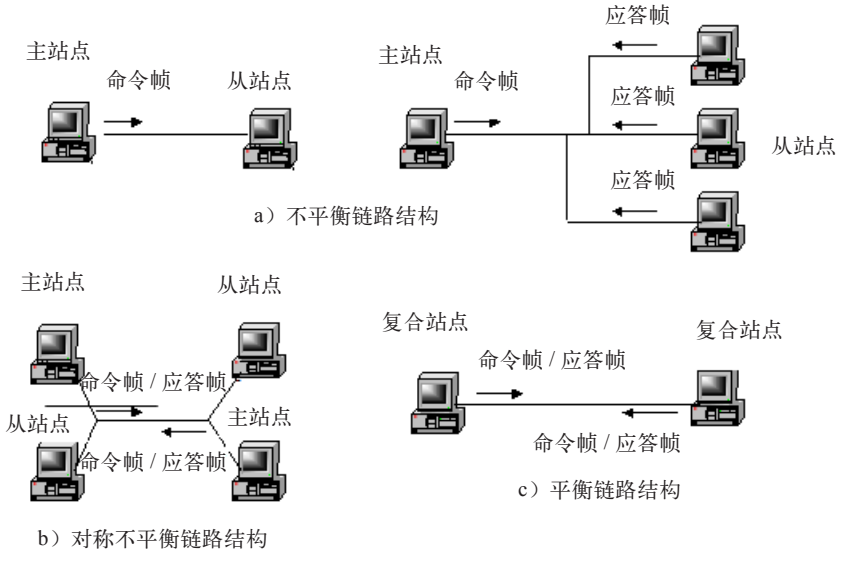


图 5-16 HDLC 的三种链路结构

根据通信双方的链路结构和传输响应类型，HDLC 提供了三种操作方式：正常响应方式、异步响应方式和异步平衡方式。

1. 正常响应方式

正常响应方式（NRM）适用于不平衡链路结构，主要用于点对点和一点对多点的链路结构中，特别是一点对多点链路，一端为主站点，另一端为从站点，如图 5-16a 所示。在这种方式中，主站点控制着整个链路的操作，负责链路的初始化、数据流控制和链路复位等，而

从站点仅可在收到主站点的明确允许后，才能发出响应，所以它们的角色是不对称的，也就是不平衡的。

2. 异步响应方式

异步响应方式（ARM）也适用于不平衡链路结构，它一般是对称不平衡链路结构，如图 5-16b 所示。ARM 与 NRM 不同的是：在 ARM 中，从站点可以在没有得到主站点允许的情况下开始数据传输，所以它的传输效率比 NRM 高。

3. 异步平衡方式

异步平衡方式（ABM）适用于平衡链路结构，即链路两端都是复合站点，也就是该站点既可作为主站点，又可作为从站点，如图 5-16c 所示。在这种链路结构中，两端的复合站点具有同等的能力，不管哪个复合站点均可在任意时间发送命令帧，并且不需要收到对方复合站点发出的命令帧就可以发送响应帧。ITU—T X.25 建议的数据链路层就采用这种方式。

除了以上三种基本操作方式外，HDLC 还有三种扩充方式，即扩充正常响应方式（SNRM）、扩充异步响应方式（SARM）、扩充异步平衡方式（SABM），它们分别与基本方式相对应。

5.6.2　SDLC/HDLC 帧结构

下面以 SDLC 和 HDLC 协议为例介绍面向比特的同步传输协议的数据帧格式，如图 5-17 所示。各字段的说明如下：

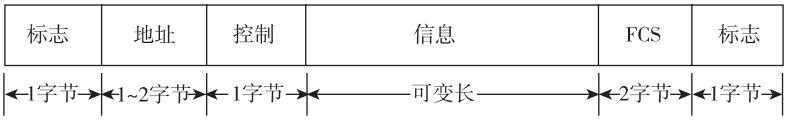


图 5-17　SDLC/HDLC 帧格式

1. 标志字段

SDLC/HDLC 协议规定，所有信息传输必须以一个“标志”字段 F（Flag）开始，且以同一个标志字段结束。也就是说每帧数据中有两个标志字段，值均为 01111110（在 EBCDIC 码中是“=”字符），占 1 字节。标志字段用于界定不同帧，以获得帧边界，实现通信双方的帧同步，因为接收端可以通过搜索“01111110”来获知帧的开头和结束。通常，在不进行帧传送的时刻，为了维持信道处于激活状态，发送端会不断地发送标志字段，使接收端认为一个新的帧传送已经开始。

SDLC/HDLC 中使用标志字段的方法可以是一个帧包括一个开始标志和一个结束标志（如图 5-18a 所示），也可以把一个帧的结束标志当成下一个帧的开始标志，也就是共享标志字段（如图 5-18b 所示），还可以把一个帧的结束标志中的最后一个“0”当成下一帧开始标志的第一个“0”（如图 5-18c 所示）。

在一串数据比特中，有可能产生与标志字段的码型相同的比特组合——01111110。为了

防止这种情况发生，保证对数据的透明传输，采取了比特填充技术。就是在信号码中出现连续 5 个“1”以后插入一个“0”，如图 5-19 所示。在接收端只需要再去掉 5 个“1”后面插入的“0”即可恢复原来的信号码序列。

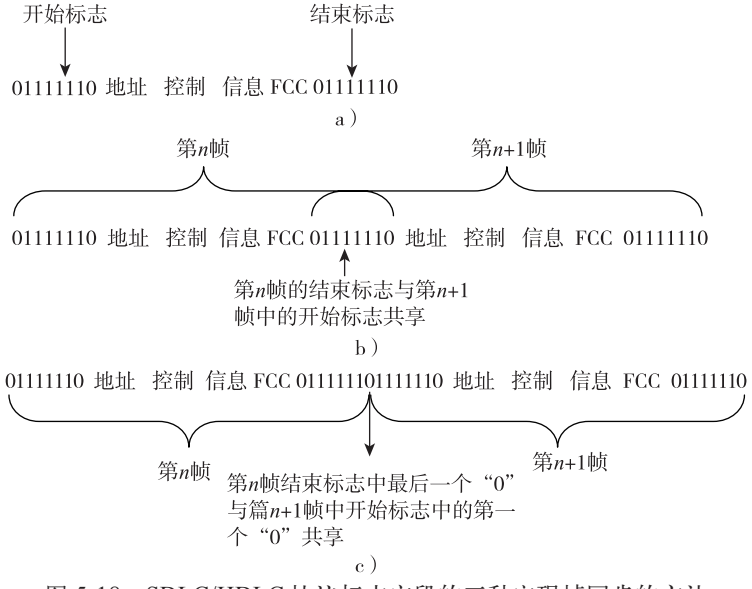


图 5-18 SDLC/HDLC 协议标志字段的三种实现帧同步的方法

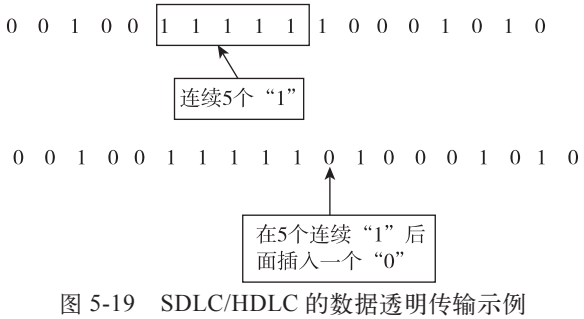


图 5-19 SDLC/HDLC 的数据透明传输示例

2. 地址字段

在“标志”字段之后是一个“地址”字段 A（Address），表明帧是来自主站点还是来自从站点。主站点或者复合站点发送的命令帧中地址字段携带的是对方从站点的地址，而从站点发出的响应帧的地址字段携带的是本站点的地址。总体来说，在使用不平衡方式（包括 NRM 和 ARM）传输数据时，地址字段总是写入从站的地址；在使用平衡方式（ABM）时，地址字段总是写入响应站点的地址。

“地址”字段可占 8 位或 16 位（也就是 1，或者 2 个字节），通常是采用 8 位长度。00000000 的地址为空地址，不能分配给任何站点，用于测试数据链路的状态；11111111 地址为

广播地址，使用这个地址就可以把一个数据帧发送到同一网段中其他所有站点。因此在采用 8 位地址格式时，总的有效地址数是 254 个，这对一般的多点链路来说是足够了。但考虑在某些情况下，例如使用分组无线网，用户可能很多，可使用扩充地址字段，以字节为单位扩充。在扩充时，每个地址字段的第 1 位用作扩充指示，即当第 1 位为“0”时，表示后续一个字节为扩充地址字段；当第 1 位为“1”时，表示后续一个字节不是扩充地址字段，地址字段到此为止。

3. 控制字段

“控制”字段 C（Control）用来实现 HDLC 协议的各种控制信息，并标志是否是数据（因为它可以标志不同的帧类型），占 1 个字节长度。发送方主站点或复合站点利用控制字段来通知被寻址的从站点或复合站点执行约定的操作；而从站点用该字段对来自主站点或复合站点的命令帧进行响应，报告已完成的操作或状态的变化。

“控制”字段的结构如图 5-20 所示，共 8 位，根据其最前面两个比特的取值，可将 HDLC 帧划分为三大类，即信息帧 I（Information）、监控帧 S（Supervisory）和无编号帧 U（Unnumbered）。如果控制字段中的第 1 位为 0，则代表发送的是 I 帧；如果控制字段中的第 1 位和第 2 位为 10，则代表发送的是 S 帧；如果控制字段中的第 1 位和第 2 位为 11，则代表发送的是 U 帧。有关这三种帧类型及各自的“控制”字段具体结构，将在本节后面介绍。

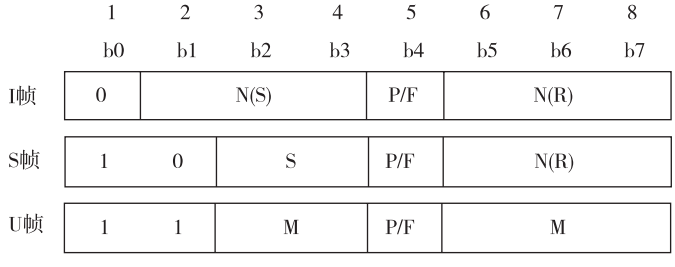


图 5-20　HDLC“控制”字段结构

在“控制”字段结构中，最难理解的就是第 5 位的 P/F（Poll/Final，查询 / 结束）位。在 HDLC 的各类帧中均带有这个 P/F 位，但在不同帧中的含义不一样：在由主站点发出的命令帧中取“P”位，起查询的作用，即当该位为 1 时，要求被查询的从站点做出响应；在从站点响应主站点的帧中取“F”位，起结束数据发送或确认结束的作用，即当该位为 1 时，表示从站点数据发送完毕，或者响应完毕。P/F 位在不同的链路结构操作方式中所代表的含义并不一样。具体如下：

在 NRM 方式中，从站点不能主动向主站点发送信息，从站点只有在收到主站点发出 P 位为 1（表示对各从站点依次进行查询，看看这些从站点是否有数据发送）的命令帧以后才能发送对应的响应帧：如果从站点有数据发送，则在最后一个 I 帧中将 F 位置 1（表示数据发送结束），中间的 I 帧 F 位置 0（表示数据还没发完）；如果从站点无数据发送，则要向主站点发送将 F 位置 1 的响应帧。

在 ARM 或 ABM 方式中，任何一个站点都可以在主动发送的 S 帧和 I 帧中将 P 位置 1（同样是表示对各从站点或者对方复合站点依次进行查询，看看它们是否有数据发送）。对方站点在收到 P=1 的 S 帧或 I 帧后，同样会按照上面介绍的方法进行处理：如果对方站点有数据发送，则在最后一个 S 帧或 I 帧中将 F 位置 1（表示数据发送结束），中间的 S 帧或 I 帧 F 位置 0（表示数据还没发完）；如果对方站点无数据发送，则要向主站点发送将 F 位置 1 的响应帧。

4. 信息字段

信息字段包含了用户的数据信息和来自上层的各种控制信息。在 I 帧和某些 U 帧中具有该字段，且可以是任意长度的比特序列。在实际应用中，其长度由收发站的缓冲器的大小和线路的差错情况决定，但必须是 8 位的整数倍。该字段可以是 0 长度，也就是可以无信息字段，如在监控帧（S 帧）中就规定不能有信息字段。

5. 帧校验序列字段

帧校验序列（FCS）字段可以使用 16 位的 CRC，对两个标志字段之间的整个帧的内容进行校验。CCITT 和 ISO 推荐使用的生成多项式为 $g(x)=x^{16}+x^{12}+x^{5}+1$ ；IBM SDLC 使用的生成多项式为 $g(x)=x^{16}+x^{15}+x^{2}+1$。 有关 CRC 的校验原理参见 5.3.2 节。

由于在 SDLC/HDLC 的帧中至少含有 A（地址）、C（控制）和 FCS（帧校验序列）这三个字段，所以整个帧长度最小为 32 位。

5.6.3 SDLC/HDLC 帧类型及其标识方法

前面说了，在 SDLC 和 HDLC 等数据链路控制协议中，有 I 帧、S 帧和 U 帧三种帧类型。下面具体介绍这些类型帧，以及在控制字段中如何标志它们。

1. 信息帧（I 帧）

I 帧用于用户数据传输，包含信息字段。在 I 帧控制字段中第 1 位固定为 0；第 2 位～第 4 位为 N（S），用于标志要发送的帧的序号；第 5 位为 P/F 位；第 6 位～第 8 位为 N（R），用于标志期待要接收的帧序号（它有两层含义：一是表示发送端已确认了前 $N-1$ 个帧，另一个是发送端期待送第 N 帧的确认帧。）

N（S）和 N（R）各占 3 位，即序号采用模 8 运算，使用 0 ～ 7 八个编号。在有些场合，如卫星通信中，模 8 已经不能满足要求了，这时可以把控制字段扩展为 2 字节，N（S）和 N（R）都可用 7 位来表示，即增加到模 128。

2. 监控帧（S 帧）

S 帧用于监视和控制数据链路，完成 I 帧的接收确认、重发请求、暂停发送请求等功能。S 帧没有信息字段（一共只有 48 位），可由主站点或从站点发送，具体要视对应的链路结构操作方式。在 S 帧的控制字段中第 1 位和第 2 位分别固定为 1、0；第 3 位和第 4 位为 S，代表监控功能；第 5 位 P/F 和第 6 ～ 8 位与 I 帧一样。

因为用于表示监控功能的 S 共有 2 位，所以可以代表 4 种监控类型的帧，具体如下：

- 00——接收就绪（RR），主站点可以使用 RR 型 S 帧来查询从站点，即希望从站点传输控制字段中第 6 ～ 8 位 N（R）所指示的编号为 N（R）的 I 帧，如果从站点存在这样的帧，便进行传输。从站点也可用 RR 型 S 帧来作为对主站点的响应帧，表示从站点希望从主站点那里接收的下一个 I 帧的编号是 N（R）。
- 01——拒绝（REJ），用于接收端要求发送端对从编号为 N（R）开始的帧及其以后所有的帧进行重发，这也暗示 N（R）以前的 I 帧已被正确接收。
- 10——接收未就绪（RNR），用于接收端通知发送端编号小于 N(R）的 I 帧已被收到，但目前正处于忙状态，尚未准备好接收编号为 N（R）的 I 帧，这可用来对链路流量进行控制。
- 11——选择拒绝（SREJ），用于接收端要求发送端发送编号为 N（R）单个 I 帧，并暗示其他编号的 I 帧已全部确认。

上面四种 S 帧中，前三种用在回退 N 帧 ARQ 方案中，最后一种只用于选择重发 ARQ 方式中。有关这两种差错控制方案参见 5.3.5 节。

3. 无编号帧（U 帧）

U 帧用于数据链路的控制，不带编号，可以在任何需要的时刻发出，而不影响带编号的信息帧的交换顺序。在 U 帧的“控制”字段中第 1 位和第 2 位都固定为 1；第 3 位和第 4 位，以及第 6 ～ 8 位均为 M，代表无编号功能；第 5 位 P/F 与 I 帧一样。

U 帧可以分为命令帧（C）和响应帧（R），通过 5 个 M 位来表示不同功能，具体如表 5-4 所示。

表 5-4　无编号帧的类型及编码

帧名称	功能	帧类型		M 位	
		命令帧	响应帧	b_2 b_3	b_5 b_6 b_7
SNRM	置正常响应模式	C		0 0	0 0 1
SARM/DM	置异步响应模式 / 断开方式	C	R	1 1	0 0 0
SABM	置异步平衡模式	C		1 1	1 0 0
SNRME	置扩充正常响应模式	C		1 1	0 1 1
SARME	置扩充异步响应模式	C		1 1	0 1 0
SABME	置扩充异步平衡模式	C		1 1	1 1 0
DISC/RD	断链 / 请求断链	C	R	0 0	0 1 0
SIM/RIM	置初始化方式 / 请求初始化方式	C		1 0	0 0 0
UP	无编号查询	C		0 0	1 0 0
UI	无编号信息	C		0 0	0 0 0
XID	交换识别	C	R	1 1	1 0 1
RESET	复位	C		1 1	0 0 1
FRMR	帧拒绝		R	1 0	0 0 1
UA	无编号确认		R	0 0	1 1 0

5.7 面向字符的 PPP 同步传输协议

前面介绍的 BSC、SDLC、HDLC 都属于局域网中的数据链路层协议，而本节要介绍的 PPP（Point-to-Point Protocol，点对点协议）是一种应用非常广泛的广域网数据链路层协议。如我们在使用 Modem 进行拨号连接时就需要用到它，路由器设备间的 Serial 口之间的连接也要封装这个协议。本节要详细介绍这一协议的工作原理以及协议结构。

5.7.1 PPP 简介

在点对点链路上，最早使用的数据链路层协议不是 PPP，而是 SLIP（Serial Line Internet Protocol，串行线路网际协议），如 Windows 98 系统中 Modem 拨号就是使用的 SLIP，而不是 PPP。但 SLIP 具有以下许多根本无法适应当时网络技术发展和应用需求的不足，具体如下：

- 连接速率低：SLIP 使用的线路速率一般介于 1200bps 和 19.2kbps 之间，远没有 PPP 的连接速率高（PPP 最高可达 128kbps）。
- 不能自动分配 IP 地址：进行 SLIP 连接的通信双方必须先具备静态 IP 地址，不能在连接过程中动态分配 IP 地址。所以当时的 SLIP 通常应用于专线连接中，在拨号连接中也仅应用于固定 IP 地址方式的连接。
- 无协议类型字段：在 SLIP 帧中没有协议类型字段，只支持默认的 IP 协议。SLIP 帧很简单，只是在 IP 包的最前面和最后面各加一个 END 字符（0xc0），作为帧边界，标志一个帧的起始和结束。如果在包中有 END 字符，则包中的 END 字符用 0xdb（ESC 字符）和 0xdc 两个字符来替代；如果在包中有 ESC 字符，则 ESC 字符用 0xdb（ESC 字符）和 0xdd 两个字符来替代，以实现数据透明传输，如图 5-21 所示。

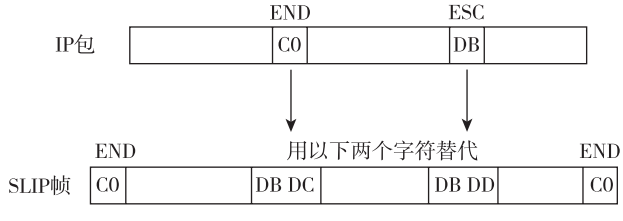


图 5-21 SLIP 帧格式及透明传输示意图

- 无帧校验序列（FCS）字段：从图 5-21 中可以看出，SLIP 帧中没有 FCS 字段，因此在 SLIP 链路层上无法检测出传输差错，必须由上层实体或具有纠错能力的 Modem 来解决差错问题。

PPP 协议是在 SLIP 的基础上发展起来的点对点数据链路层协议，对 SLIP 以上问题均加以了解决，但仍是面向字符的链路层协议。具体来讲，PPP 协议具有以下几个方面的特性：

- PPP 在连接速率上可以远高于 SLIP，可以最高达到 128kbps，如像 v.90 以上标准的

Modem 都可达到 64kbps。

- 提供了协议类型字段和帧校验序列（FCS）字段（如图 5-22 所示），使得 PPP 除了支持 IP 协议包封装外，还可以封装其他三层协议包，如当时 DECnet 和 Novell 的 Internet 网包交换（IPX）。另外有了 FCS 字段，可以提供各种差错控制功能。

8	8	8	16	可变	16 ～ 32	8 位
标志	地址	控制	协议	信息	FCS	标志

图 5-22　PPP 帧格式

- 提供了一整套方案来解决链路建立、维护、拆除、上层协议协商、认证等问题。这些功能是通过以下几个子协议来完成的：
  - 链路控制协议（Link Control Protocol，LCP）：用于建立、配置、测试和管理数据链路连接。
  - 网络控制协议（Network Control Protocol，NCP）：协商该链路上所传输的数据包格式与类型，建立、配置不同的网络层协议。
  - 口令认证协议（Password Authentication Protocol，PAP）和质询握手认证协议（Challenge-Handshake Authentication Protocol，CHAP）：为 PPP 连接提供用户认证功能，可以确保 PPP 连接的安全性。

家庭拨号上网就是通过 PPP 在用户端和运营商的接入服务器之间建立通信链路。在宽带接入技术日新月异的今天，PPP 协议也衍生出新的应用。典型的应用是在 ADSL（Asymmetrical Digital Subscriber Loop，非对称数据用户环线）接入方式当中，PPP 协议与其他的协议共同派生出了符合宽带接入要求的新的协议，如 PPPoE（PPP over Ethernet），PPPoA（PPP over ATM）。

利用以太网（Ethernet）资源，在以太网上运行 PPP 来进行用户认证接入的方式称为 PPPoE。PPPoE 既保护了用户方的以太网资源，又完成了 ADSL 的接入要求，是目前 ADSL 接入方式中应用最广泛的技术标准。同样，在下面将要介绍的 ATM（异步传输模式，Asynchronous Transfer Mode）网络上运行 PPP 协议来管理用户认证的方式称为 PPPoA。它与 PPPoE 的原理相同，作用相同；不同的是它在 ATM 网络上运行，而 PPPoE 在以太网网络上运行，所以要分别适应 ATM 标准和以太网标准。

PPP 协议简单和功能完整的特点使它得到了广泛的应用，相信在未来的网络技术发展中，还可以发挥更大的作用。

5.7.2　PPP 帧结构和透明传输原理

比较图 5-22 所示的 PPP 帧结构和图 5-17 所示的 HDLC 帧结构可以看出，它们是很相似的，其主要区别是 PPP 帧结构中多了一个协议字段（用来所封装的三层协议包类型），而且

PPP 是面向字符的协议，而 HDLC 是面向比特的协议，所以自然它们所采用的填充方式也不一样。下面先介绍 PPP 帧结构中各字段。

1. PPP 帧结构

PPP 帧结构参见图 5-22，共分 7 个字段，其中标志字段在帧的最前面和最后面均有一个，其他字段各一个。下面是这些字段的具体含义说明。

- 标志（Flag）：用来标志帧的起始或结束，占 8 位（1 个字节），值固定为 01111110（0x7E），与 HDLC 帧中的标志字段的值是一样的。
- 地址（Address）：本来是用来标志对方节点地址的，但因 PPP 是点对点通信协议，是明确知道对方节点的，在实际通信中是无须知道对方的数据链路层地址（也就是 MAC 地址），从实际通信角度考虑，此地址字段实际上是没什么意义的，所以在 PPP 帧中此地址字段为固定的 11111111（0xFF）标准广播地址，占 8 位（1 个字节）。这与 HDLC 中的地址字段是不一样的。
- 控制（Control）：因为 PPP 本身是一种可靠的点对点数据路层通信协议，无须像 HDLC 协议那样需要额外提供可靠的链路连接服务，所以 PPP 只有一种帧类型，那就是 UI（无编号信息）帧。又因为它是无编号的帧，也就是无须接收端对收到的帧进行确认，所以，PPP 帧中的控制字段其实也没有意义，值固定为 00000011（0x03）。
- 协议（Protocol）：PPP 帧与 HDLC 帧的最大区别就是 PPP 帧中有协议字段，而 HDLC 帧中无该字段。之所以 PPP 帧中有协议字段，是因为它除了可以封装 IP 协议外，还可封装其他多种网络层协议包，如 IPX、AppleTalk 等。协议字段占 16 位（2 个字节），指示在信息字段中封装的数据类型，如 0x0021 表示信息字段是 IP 数据包，0xC021 表示信息字段是 LCP（链路控制协议）数据，0x8021 表示信息字段是 NCP（网络控制协议）数据包，0xC023 表示信息字段是 PAP 安全性认证数据包，0xC223 表示信息字段是 CHAP 安全性认证数据包，0x0029 表示信息字段为 Apple Talk 协议数据包，……
- 信息（Information）：来自上层（“网络层”）的有效数据，可以是任意长度，默认为 1500 字节，如果不够该长度，还可以通过填充方法达到这个长度。
- 帧校验序列（FCS）：使用 16 位的循环冗余校验计算信息字段中的校验和，以认证数据的正确性。

2. 透明传输

从前面介绍的 PPP 帧结构中可以看出，在帧的首尾均有一个用于标志帧边界的标志字段，其值均固定为 01111110（0x7E），这就同样要面对一个问题，那就是当在信息字段中出现和标志字段一样的比特 0x7E 时，接收端可能误把这些位当成帧边界。为了解决这个问题，也就是实现透明的数据传输，就必须采取一些措施。但因为 PPP 是面向字符协议，所以它不能采用 HDLC 所使用的零比特插入法，而是使用一种特殊的字符（也就是前面所说的转义字

符）——0x7D 进行填充。具体的做法是将信息字段中出现的每一个 0x7E 字节转变成 2 字节序列（0x7D，0x5E），即 01111101 01011110；如果信息字段中出现一个 0x7D 的字节，则要将其转变成 2 字节序列（0x7D，0x5D）即 01111101 01011101；如果信息字段中出现 ASCII 码的控制字符（如值为 0x27 的 ESC 字符），则在该字符前面要加入一个 0x7D 字节，如图 5-23 所示。这样做的目的是防止这些表面上的 ASCII 码控制字符被错误地解释为控制字符。

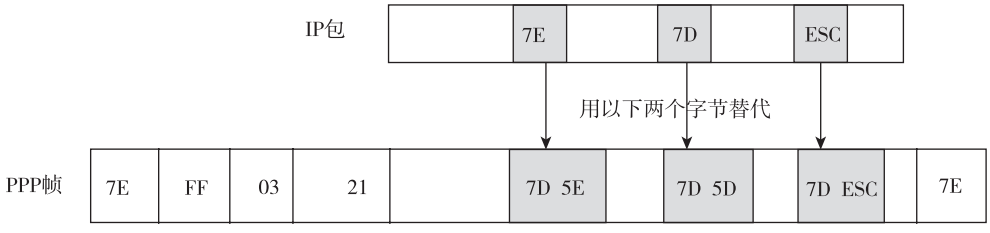


图 5-23　PPP 帧格式及透明传输示意图

5.7.3　PPP 链路建立、使用和拆除流程

在 PPP 通信中，因为不是像局域网中的链路那样始终连接的，所以在建立 PPP 通信前，通信双方必须协商建立链路连接，在链路建立后才可进行数据传输，数据传输完成后又可拆除原来建立的链路。整个过程分为五个阶段，即 Dead（死亡）阶段、Establish（链路建立）阶段、Authenticate（身份认证）阶段、Network（网络控制协商）阶段和 Terminate（结束）阶段。不同阶段进行不同协议的协商，只有前面的协议协商出结果后，才能转入下一个阶段协议的协商，如图 5-24 所示。

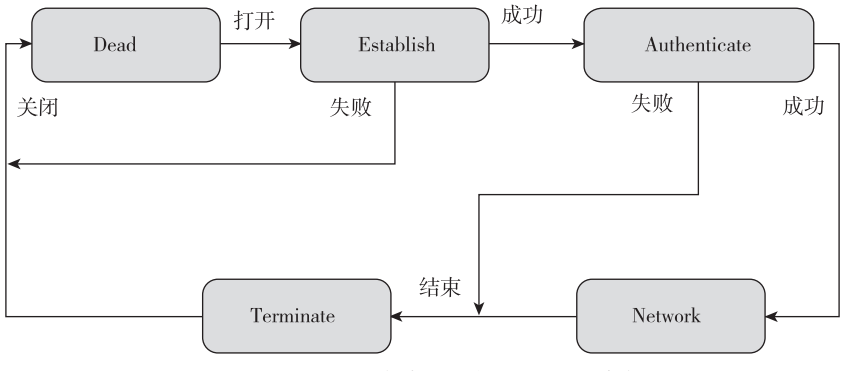


图 5-24　PPP 链路建立、使用和拆除流程

PPP 链路建立、使用和拆除的具体流程如下：

1）当有用户向 ISP 或者对端节点发起 PPP 连接请求时，首先打开物理接口，然后 PPP 在建立链路之前先通过封装了 LCP 的 PPP 帧与接口进行协商，协商内容包括工作方式是 SP（单 PPP 通信）还是 MP（多 PPP 通信）、认证方式和最大传输单元等。

2）LCP 协商完成后就进入 Establish 阶段，进行数据链路的建立。这时主要是启用 PPP 数据链路层协议，对接口进行封装。如果启用成功，则进入下一身份认证（Authenticate）阶段，并保持 LCP 为激活状态，否则返回关闭接口，LCP 的状态为关闭。

3）如果数据链路建立成功，则进入到 Authenticate 阶段，对请求连接的用户进行身份认证。具体要根据通信双方所配置的身份认证方式来确定是采用 CHAP 还是 PAP 身份认证。

4）如果认证成功就进入 Network 阶段，使用封装了 NCP 的 PPP 帧与对应的网络层协议进行协商，并为用户分配一个临时的网络层地址（如 IP 地址）；如果身份认证失败，则直接进入 Terminate（结束）阶段，拆除链路，返回到 Dead 阶段，LCP 状态转为 Down。

5）PPP 链路将一直保持通信，直至有明确的 LCP 或 NCP 帧关闭这条链路，或发生了某些外部事件（如用户的干预），进入到 Terminate 阶段，然后关闭 NCP 协议，释放原来为用户分配的临时网络层地址，最后返回到 Dead 阶段，关闭 LCP。

5.7.4 PPP 的 PAP/CHAP 身份认证

在 PPP 通信中，可以采用 PAP（密码认证协议）或者 CHAP（质询握手认证协议）身份认证方式对连接用户进行身份认证，以防非法用户的 PPP 连接。如果双方达成一致，也可以不采用任何身份认证方式（如一般情况下的路由器间 Serial 口之间的 PPP 连接）。

1. PAP 身份认证

PAP 身份认证过程非常简单，是一个二次握手机制，整个认证过程仅需两个步骤：被认证方发送认证请求→认证方给出认证结果。

PAP 身份认证可以在一方进行，即由一方认证另一方身份，也可以进行双向身份认证，也就是既要 PAP 服务器对 PAP 客户端的合法性进行认证，PAP 客户端也需要对 PAP 服务器进行认证，以确保用于认证的 PAP 服务器是合法的。如果是双向认证，则要求认证的双方都要通过对方的认证程序，否则无法在双方之间建立通信链路。

下面以单向认证为例介绍 PAP 认证过程，如图 5-25 所示。但要注意的是，PAP 认证是由被认证方（也就是 PAP 客户端）首先发起的。

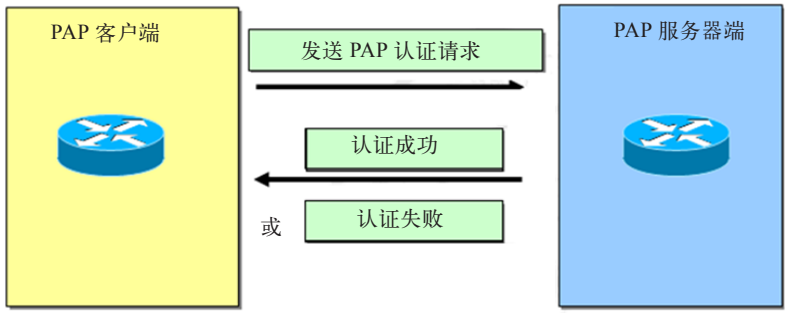


图 5-25　PAP 身份认证的两次握手

1）发起 PPP 连接的客户端（被认证方）首先向担当身份认证的 PAP 服务器端（如是向 ISP 拨号，则 PAP 服务器在 ISP 端，如果是路由器的串口对连，则 PAP 服务器必须要在对端设备上配置）发送一个认证请求帧，其中就包括用于身份认证的用户名和密码。

2）PAP 服务器端（认证方）在收到客户端发来的认证请求帧后，先查看服务器本地配置的用户账户数据库，看是否有客户端提供的用户名 / 密码对信息（这个用户账户数据库存必须先在 PAP 服务器端配置好）。如果有，则表明客户端具有合法的用户账户信息，向 PAP 客户端返回一个认证确认（ACK）帧，表示认证成功，该用户可以与 PAP 服务器端建立 PPP 连接；否则返回一个认证否认（NAK）帧，表示认证失败，当然客户端也就不能与 PAP 服务器端建立 PPP 连接了。但这里要注意的是，如果第一次认证失败，并不会马上直接将链路关闭，而是会在 PAP 客户端提示可以尝试以新的用户账户信息进行再次认证，只有当认证不通过次数达到一定值（默认为 4）时才会关闭链路，以防止因误传、网络干扰等造成不必要的 LCP 重新协商过程。

PAP 身份认证的特点是，用于身份认证的用户名及密码在网络上是以明文（也就是不加密）方式进行传输的，如在传输过程中被截获，便有可能对网络安全造成极大的威胁，所以 PAP 并不是一种安全有效的认证方法。

以上介绍的是 PAP 单向认证过程，仅两步（一问一答的形式），很简单。PAP 双向认证过程与单向认证过程类似，只不过此时 PPP 链路的两端同时具有客户端和服务器双重角色，任何一端都可向对方发送认证请求，同时对对方发来的认证请求进行认证。

2. CHAP 身份认证

CHAP 认证过程相对前面介绍的 PAP 认证来说更为复杂，它采用的是三次握手机制（而不是 PAP 中的两次握手机制），整个认证过程要经过三个主要步骤：认证方要求被认证方提供认证信息→被认证方提供认证信息→认证方给出认证结果。

另外，CHAP 身份认证方式相对 PAP 认证方式来说更加安全，因为在认证过程中，用于认证的用户名和密码不是直接以明文方式在网络上传输的，而是经过 MD5 之类的摘要加密协议随机产生的密钥；而且这个密钥是有时效的，原密钥失效后会随机产生新的密钥，所以即使在通信过程中密钥被非法用户破解了，也不会适用于后面的通信截取。

与同 PAP 认证一样，CHAP 认证也可以是单向或者双向的。如果是双向认证，则要求通信双方均要通过对对方请求的认证，否则无法在双方建立 PPP 链路。在此，我们仍以单向认证为例介绍 CHAP 认证流程，具体如图 5-26 所示。注意，CHAP 身份认证首先是由 CHAP 服务器端发起的。

1）当 PPP 链路建立起来后，采用 CPAP 身份认证方式时，首先是由 CHAP 服务器（认证方）不断地以产生一个随机序列号的质询（challenge，又称挑战）字符串帧发送给 CHAP 客户端（被认证方），询问客户端（被认证方）是否要进行身份认证。直到该客户端为这个质询做出了响应，也就是进入了下面的第 2 步。

2）客户端在收到服务器端发来的质询消息后，把自己要用于身份认证的用户名和密码

（这需要事先在客户端设备中配置好）通过 MD5 摘要加密协议生成一个随机序列的响应帧发送给服务器端。

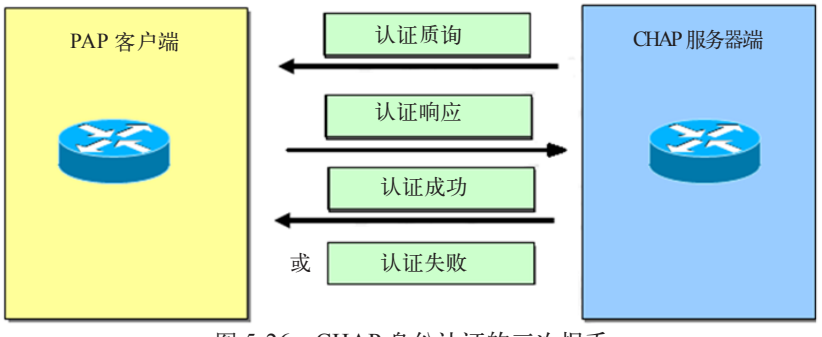


图 5-26　CHAP 身份认证的三次握手

3）服务器端在收到来自客户端的响应后，同样利用 MD5 加密协议解密出其中的认证用户名和密码，然后在服务器本地用户账户数据库（也是需要事先在服务器端配置好的）中查找，看是否有相同的账户信息。如果找到一样的账户信息，表明请求认证的客户端是合法的，通过认证，允许客户端发起的 PPP 连接，否则拒绝认证，表示认证失败。与 PAP 一样，第一次认证失败后，也不会马上关闭链路，而是再次向客户端提示输入新的用户名和密码进行再次认证，直到规定的最高尝试次数。

CHAP 认证方式的最关键一点就是采用了 MD5 摘要加密协议，用于认证的用户名和密码是直接在摘要消息中经过加密的，所以不会在网络中以明文方式传输，因此它的安全性要比 PAP 高。

5.8　数据链路层主要网络设备

介绍完数据链路层主要功能、服务后，下面我们再来简单介绍工作在数据链路层上的主要网络设备。在常见的网络设备中，如网卡、网桥和二层交换机。

5.8.1　计算机网卡

说到计算机网卡（也叫网络适配器），大家可能认为这没什么好说的，大家都见过、用过。网卡是安装在计算机上，用来连接计算机网络的，是计算机网络中最基础的网络设备。目前在计算机局域网中，网卡类型总的来说主要可分为有线以太网卡、WLAN 无线网卡两大类。下面分别予以介绍。

1. 有线以太网卡

在企业局域网中，我们通常所说的有线网卡通常就是指以太网卡，所以在此也仅限于对以太网卡的介绍，而不再涉及目前在企业局域网中很少见到的其他网络类型的网卡，如令牌

环网卡、令牌总线网卡和 FDDI 网卡等。另外，网卡除了要区分网络类型外，还可根据所应用的环境分为普通工作站网卡和服务器网卡两类。

有线网卡还可根据以下几个方面可以进行进一步划分：一是网卡的主机接口，也就是网卡与计算机的接口；二是网卡主机接口总线的位数；三是网卡的网络接口类型，也对应了网卡所支持的传输介质类型；四是网卡所支持的以太网标准。

在主机接口方面，有线网卡主要有以下几种类型：一是最常见的 PCI 总线接口，二是微型 PCI（PCMCIA）接口，三是在服务器网卡用得比较多的 PCI−X 和 PCI−E 接口。这几种主机接口类型网卡如图 5-27 所示。

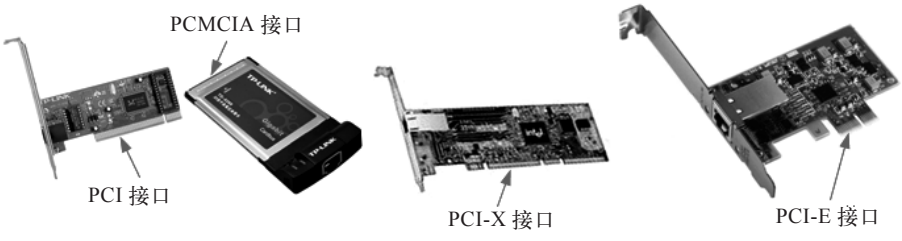


图 5-27　PCI、PCMCIA、PCI-X 和 PCI-E 接口网卡

在有线网卡的网络接口方面，主要对应的是不同的传输介质。因为使用同轴电缆的以太网目前在企业局域网中比较少见了，所以在此不再介绍粗 / 细同轴电缆接口的以太网卡，仅介绍双绞线接口以太网卡和光纤接口以太网卡。

双绞线接口以太网卡使用最普遍，对应双绞网线连接器的接口为 RJ-45 类型。RJ-45 连接器（俗称水晶头）和网卡上对应的 RJ-45 网络接口如图 5-28 所示。

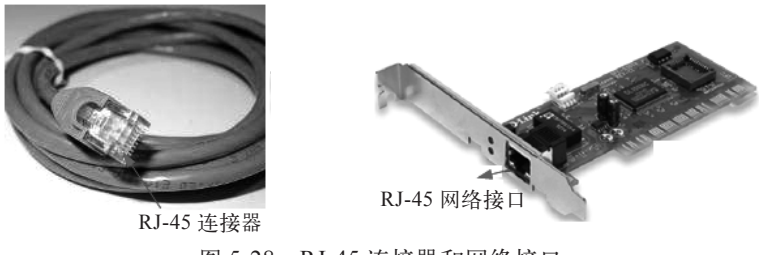


图 5-28　RJ-45 连接器和网络接口

使用光纤作为传输介质的网卡所对应的光纤连接器接口有好几种，最常见的有 ST、SC、FC、LC 四种。这在本书第 4 章中有详细介绍，在此不再赘述。ST、SC、FC 和 LC 这四种主要的光纤连接器接口如图 5-29 所示。图 5-30 所示是目前应用最广的两种光纤网络接口 SC 和 LC 的网卡示例。

有线网卡除了可以按主机接口和网络接口划分外，还可以根据网卡所支持的网络标准来划分。目前，在有线以太网工作站中我们通常采用支持 10/100Mbps 自适应的双绞线快速

以太网卡，性能要求高一些的可能会用到支持自适应的 10/100/1000Mbps 双绞线千兆以太网卡。而服务器上的网卡目前基本上都是自适应的 10/100/1000Mbps 双绞线千兆以太网卡，或者纯 1000Mbps 的光纤千兆以太网卡。

图 5-29　ST、SC、FC 和 LC 光纤连接器接口

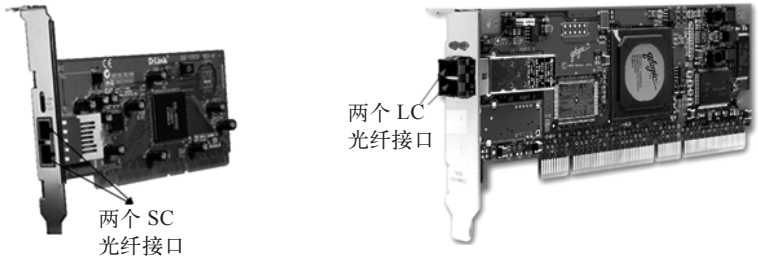
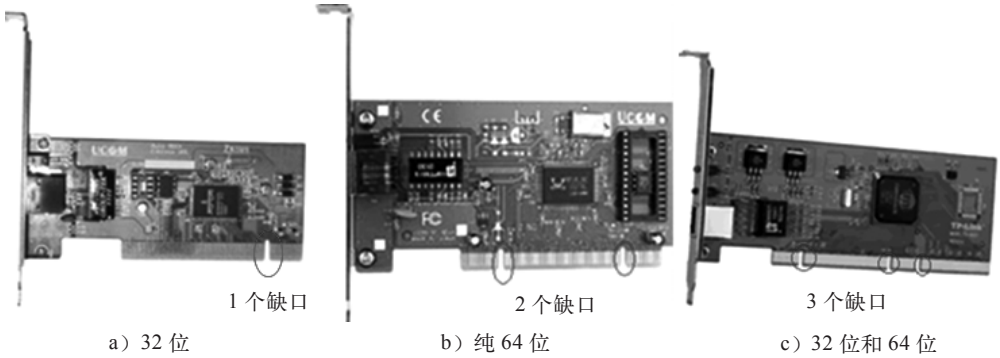


图 5-30　SC 和 LC 网络接口网卡示例

另外，支持的这些不同网络标准的 PCI 接口以太网卡，所对应的主机接口工作模式也会有所不同。工作站上普遍使用的 PCI 接口快速以太网卡基本上都是 32 位的，而千兆以太网卡基本上都是 64 位的。要注意的是，纯 64 位的千兆以太网卡与向后兼容 32 位的 64 位以太网卡的 PCI 总线接口金手指结构（金手指长度和缺口数不一样）是不一样的。如图 5-31a 所示是 32 位的 PCI 接口快速以太网卡，图 5-31b 所示为纯 64 位的 PCI 接口千兆以太网卡，图 5-31c 为同时支持 32 位和 64 位的 PCI 接口千兆以太网卡。



a）32 位　　b）纯 64 位　　c）32 位和 64 位

图 5-31　三种不同工作模式的以太网卡

2. WLAN 网卡

在 WLAN 网卡方面，相对有线网卡来说要简单一些，那是因为就目前来说 WLAN 技术

还不如有线网络那么先进，所以在主机接口方面没有什么特殊要求，一般的 32 位 PCI 或者 USB 1.1、2.0 或者 3.0 版本接口即可满足。

在 WLAN 网卡选择方面主要考虑网卡的主机接口和所使用的 WLAN 技术标准。在台式机工作站中通常选用 PCI 或者 USB 接口的 WLAN 网卡（主要是 PCI 接口），如图 5-32 所示。

a）PCI　　b）USB

图 5-32　PCI 和 USB 接口的 WLAN 网卡

对于笔记本电脑用户则可以选择 PCIMCIA 和 USB 两种接口类型的无线局域网网卡。而 PCMCIA 接口又分 16 位的 PCMCIA 和 32 位的 CardBus 两种接口类型，如图 5-33 所示。在无线局域网标准上，目前至少建议应选择具有 54Mbps 速率的 IEEE 802.11g 标准无线网卡产品，普遍采用的是支持 600Mbps 速率的 IEEE 802.11n 标准的无线网卡。有关 WLAN 标准的内容将在下章介绍。

a）PCMCIA　　b）CardBus

图 5-33　PCMCIA 和 CardBus 接口 WLAN 网卡

5.8.2　网桥及其工作原理

网桥（Bridge）是早期的两端口二层网络设备，用来连接不同网段的计算机网络设备（如图 5-34 所示），同时它又可隔离冲突域，因为它的两个端口不是共享一条背板总线（分别有一条独立的交换信道），比当时的集线器（Hub）性能更好（集线器上各端口都是共享同一条背板总线的）。后来，网桥被具有更多端口、同时也可隔离冲突域的交换机（Switch）所取代。

1. 理解“网桥”的含义

也有人把“网桥”比喻成一个聪明的中继器（Repeater）。因为中继器只是对所接收的信号进行放大，然后直接发送到另一个端口连接的电缆上，主要用于扩展网络的物理连接范

围；而网桥除了可以扩展网络的物理连接范围外，还可以对 MAC 地址进行分区，隔离不同物理网段之间的碰撞（也就是隔离“冲突域”）。集线器和中继器都是物理层设备，而网桥属于二层设备。

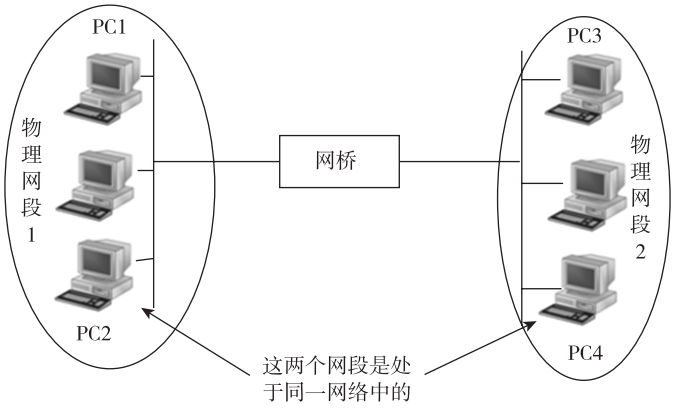


图 5-34　网桥连接的两个物理网段

我们经常听到这样的说法，那就是“网桥”是一种可连接不同网段的二层网络设备（二层交换机也一样），一个端口可以连接一个网段。所以很多人不总在纳闷，网桥怎么能连接不同网段呢？其实这是因为大家对这里所说的“网段”并不理解。其实这里“网段”更准确地讲应该是叫“物理网段”，是指 IP 地址属于同一网络地址段（也就是 IP 地址中的网络 ID 一样），位于不同地理位置的不同 LAN 分段，是基于物理意义上的地理区域进行划分的。我们常说的网段是指 IP 地址属于不同网络地址段的网络或子网，是一个三层概念，其实这应该叫做逻辑网段，是基于逻辑意义上的网络地址进行划分的。

无论是网桥，还是二层交换机，虽然每个端口可以连接一个网段，但是它们所连接的主机都在同一网络，或者同一子网中。如连接的主机位于不同办公室或者不同办公楼中，则可采用同一网络地址的两个或多个小 LAN，以组成一个可以统一管理的大 LAN。但要注意的是，因为网桥只有两个端口，所以所连接的两个物理网段的主机通常就是由当时的集线器进行集中连接的（网桥端口通常不是直接连接主机的）。软件中通常所说的桥接（如 VMware 中的桥接工作模式）也就是网桥的作用，它连接的也是同一网络或子网中的两个网段。

2. 网桥工作原理解析

前面说到了网桥具有两种主要特性：一是可基于物理网段的 MAC 地址进行学习，二是可以隔离冲突域。下面通过一个示例来进行解析。

假设图 5-34 中所示的物理网段 1 和物理网段 2 中的主机都是通过集线器集中连接的，则这样这两个物理网段各自形成一个冲突域，因为集线器是采用共享介质传输的，而网桥的背板信道不是共享的（每个端口的数据收发都有一条单独的信道），所以一个集线器就是一个冲突域。网桥的数据转发原理如图 5-35 所示。下面是具体的解析。

说明　MAC 地址表也就是通常所说的 CAM（Content Addressable Memory，内容可寻址存储器）表，保存的是对应 MAC 地址主机与所连接的交换机端口的映射。这个映射表项可以由管理员手动绑定创建，也可以由交换机自动学习得到。在交换机上可以通过一些命令（如 Cisco 交换机是使用 show mac-address-table 命令）查看。下面是一个在交换机上查看 MAC 地址和端口映射表的示例，其中列出了交换机中为 CPU 分配的静态（static）MAC 地址和通过学习功能自动学习得到的动态（dynamic）MAC 地址，其中的 Ports 列显示的是对应 MAC 地址主机所连接的端口，VLAN 列则为对应主机连接端口所属的 VLAN。

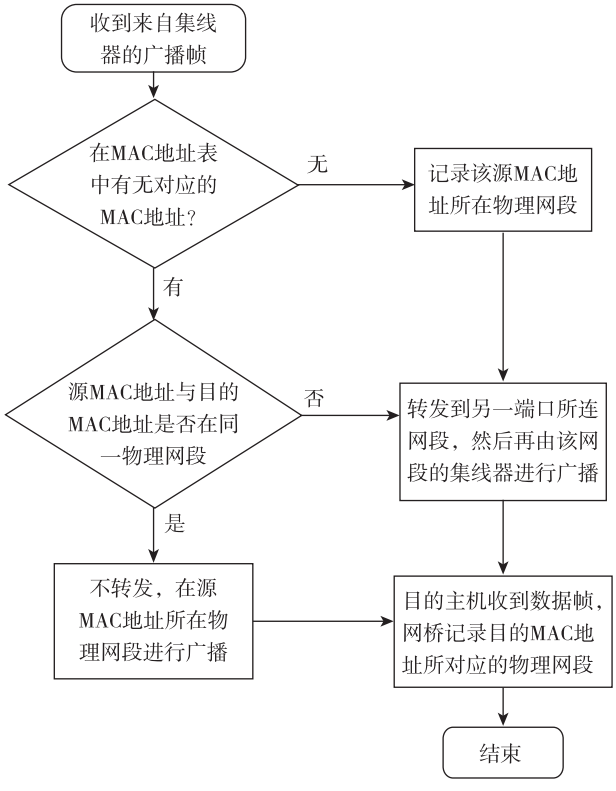


图 5-35　网桥数据转发原理示意图

```
switch#show mac-address-table
          Mac Address Table
   ------------------------------------------
Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
 All    0100.0ccc.cccc    STATIC      CPU
 All    0100.0ccc.cccd    STATIC      CPU
 All    ffff.ffff.ffff        STATIC      CPU
  1    0000.0c07.accb    DYNAMIC     Gi0/1
  1    0002.8501.de00    DYNAMIC     Gi0/1
```

```
  1    0015.f915.8e80    DYNAMIC     Gi0/1
  1    0016.7694.c009    DYNAMIC     Gi0/1
  1    0020.ed14.399c    DYNAMIC     Gi0/1
  1    0030.b637.8e10    DYNAMIC     Gi0/1
  1    0050.ba10.404a    DYNAMIC     Gi0/1
100     0007.847b.c40a    DYNAMIC     Gi0/1
100     00d0.d3a4.7cec    DYNAMIC     Gi0/1
110     0006.28bb.71c0    DYNAMIC     Gi0/1
110     00d0.d3a4.7cec    DYNAMIC     Gi0/1
120     0000.b497.8250    DYNAMIC     Fa0/20
120     0002.b3d8.68e7    DYNAMIC     Fa0/20
120     0002.b3d8.6928    DYNAMIC     Fa0/20
120     0003.a03a.03fc    DYNAMIC     Fa0/19
```

现假设图 5-34 所示网络中的一台 PC 要向另一台 PC 发送数据。因为集线器也是物理层设备，不能识别帧中的 MAC 地址，所以无论是哪台主机要发送数据，在集线器上都是以广播方式进行的，连接该集线器上的所有节点都会收到这个广播帧，包括网桥连接到该集线器的端口。

1）当网桥收到集线器的广播帧后，网桥会把帧中的源 MAC 地址和目的 MAC 地址与网桥缓存中保存的 MAC 地址表进行比较。

2）最初，网桥的缓存中是没有任何 MAC 地址的，所以一开始它也不知道哪台主机在哪个物理网段上，收到的所有帧都直接以泛洪方式（也是复制原数据帧）转发到另一个端口上，同时会把数据帧中的源 MAC 地址所对应的物理网段记录下来（其实就是与对应的网桥端口对应起来）。

3）在数据帧被某个 PC 机接收后，也会把对应目的 MAC 地址所对应的物理网段记录在缓存中的 MAC 表中。这样，经过多次这样的记录，就可以在 MAC 地址表中把整个网络中各主机 MAC 地址与对应的物理网段全部记录下来。因为网桥的端口通常是连接集线器的，所以一个网桥端口会与多个主机 MAC 地址进行映射。

4）当网桥收到的数据帧中源 MAC 地址和目的 MAC 地址都在网桥 MAC 地址表中可以找到时，网桥会比较这两个 MAC 地址是否属于同一个物理网段。如果是同一物理网段，则网桥不会把该帧转发到下一个端口，直接丢弃，起到冲突域隔离作用。相反，如果两个 MAC 地址不在同一物理网段，则网桥会把从一个物理网段发来的帧转发到连接另一个物理网段上，然后再通过所连接的集线器进行复制方式的广播。

5.8.3 二层交换机概述

在计算机网络设备还有一种是工作在数据链路层的，那就是二层交换机（其实三层或以上层次交换机同样提供二层交换功能）。

交换机（Switch）可以说同时是集线器和网桥的升级换代产品，因为交换机具有集线器一样的集中连接功能，同时它又具有网桥的数据交换功能。所以可以这样说，交换机是带有

交换功能的集线器，或者说交换机是多端口的网桥。外形上，集线器与交换机产品没什么太大区别，如图 5-36a 所示为一款集线器产品，而图 5-36b 所示为一款交换机产品。

a）集线器

b）交换机

图 5-36　集线器与交换机的外观比较

从图中的对比可以看出，交换机与集线器一样，是一个具有许多同类端口的网络设备。当然图中的对比仅能起到一般意义上的外观比较，实际上，因为交换机发展相当快，其应用向两个不同的方向发展，所以在外观上也有很大的区别。小的桌面交换机就像我们现在用的 Modem 一般大（集线器也有这样小的），而大的则采用模块化结构，机箱较大，而不是像图中显示那样像一个长方形盒。图 5-37 所示代表小型固定端口的桌面型交换机和大型模块化交换机。

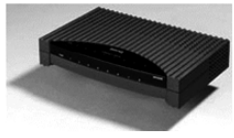


图 5-37　小型固定端口交换机与模块化交换机比较

1. 交换机的主要特性

交换机（此处特指二层交换机）是上节介绍的网桥的升级产品，同样是工作在网络体系结构中第二层（数据链路层）。但它与网桥相比又具有一些特性，具体体现在以下几个方面：

（1）具有多个交换端口

我们知道网桥通常只是两个交换端口，其设计目的主要就是用来连接两个距离超过单段网线传输限制的物理网段（当然也可以用来直接连接两台主机），所以它的应用受到比较多的限制。再加上当时用于主机和其他网络设备集中连接的设备仍是传输效率和信道利用率都非常低下的集线器，根本不适应于计算机网络的发展。有了交换机后，一台交换机可以有多个端口，而且与网桥一样，不仅每个端口可以连接一个不同的物理网段（交换机上一个端口对应一个物理网段），还可以有大量的端口来集中连接主机，这时交换机就可以同时担当集线器和网桥的双重角色，而且在使用性能和扩展性能、交换性能等方面都有较大提高，大大促进了计算机网络的发展。

（2）数据转发效率更高

在网桥时代，集中连接主机的仍是集线器，而我们知道集线器发送数据是采用广播方式，所以信道中的无效载荷比例相当高，造成数据转发率和信道利用率都非常低。而有了交换机后，因为大多数主机都是直接连接在交换机端口上，即使不是，也主要是连接在其他交换机端口，所以数据的转发基本上都是通过提取帧中的 MAC 地址直接发送到目的主机上的，而不是通过广播方式（仅在未知目的 MAC 地址时采用广播），数据转发效率和信道利用率都大幅提高。

（3）更强的 MAC 地址自动学习能力

我们知道，网桥通常只有两个端口，仅可以连接两个由集线器集中连接的物理网段，所以它的 MAC 地址自动学习功能仅限于它的两个端口与对应的物理网段的映射。这样就造成了，一个网桥端口要与多个源主机 MAC 地址之间的映射，也就是一对多映射关系。而交换机上的端口多数是直接连接主机的，所以在映射表中基本上都是一个源主机 MAC 地址与一个交换端口间的一对一映射。一对一的映射查找起来明显比一对多的映射效率要高，所以交换机在数据转发效率要高于网桥。另外，交换机的缓存通常比网桥的要大，所以交换机中可以保存的 MAC 地址与端口映射表较多，更适用于较大网络。

2. 交换机与集线器的区别

交换机最开始是为了解决集线器共享传输介质、端口带宽过窄、容易产生广播风暴而产生的。最初的交换机是工作在 OSI 开放体系结构中的第二层，所以又称二层交换机。交换机与集线器的区别主要体现在如下几个方面：

（1）在 OSI 中的工作层次不同

交换机（特指二层交换机，下同）和集线器在 OSI 开放体系模型中对应的层次不一样，集线器工作在第一层（物理层），而交换机至少是工作在第二层，更高级的交换机可以工作在第三层（网络层）、第四层（传输层）和第七层（应用层），对应也就有三层交换机、四层交换机、七层交换机等之说了。本章仅介绍二层交换机。

（2）数据传输方式不同

集线器的数据传输方式是多次复制方式的广播传输，而交换机的数据传输是有目的的，数据只对目的节点发送，只是在自己的 MAC 地址表中找不到的情况下第一次使用以 FF-FF-FF-FF-FF-FF 作为 MAC 地址的“泛洪”广播方式传输。所以，交换机在数据传输效率和信道利用率方面要远高于集线器，集线器更容易产生“广播风暴”。

（3）背板信道占用方式不同

在带宽占用方面，集线器所有端口都是共享集线器背板中的一条信道带宽，而交换机的每个端口的收、发都有独享的背板信道带宽，属于“交换”方式。这样一来更进一步使得交换机的数据传输效率以及传输性能要远高于集线器

（4）数据通信方式不同

集线器因为是所有端口共享一条背板信道，所以只能采用半双工方式进行传输，同一时

间，要么是接收数据，要么是发送数据。而交换机中各端口的信道是采用矩阵交换方式，可以同时进行数据交换，也就是可以进行全双工数据传输。这也使得交换机比集线器的数据通信效率要远高于集线器。

3. 交换机的分类

性能越强的设备，应用越广，这也导致了该设备的技术不断发展，基于各种不同应用的设备类型也越多。交换机就是这样一种应用非常广泛的网络设备，它自最初出现至今，各种不同的交换机类型不断涌现，令人目不暇接。下面介绍一些目前仍广泛应用的交换机分类方式。

（1）根据网络类型划分

根据交换机所应用的局域网类型可以将局域网交换机分为标准以太网交换机（10Mbps 传输速率）、快速以太网交换机（100Mbps 传输速率）、千兆以太网交换机（1000Mbps 传输速率）、十千兆以太网交换机（10000Mbps 传输速率）等。

（2）按交换机结构划分

如果按交换机结构来划分的话，交换机可分为固定端口交换机和模块化交换机两种。固定端口顾名思义就是它所带有的端口是固定的。例如，如果是 8 端口的，那就最多只能接 8 个设备，不能再添加；如果是 16 个端口的也就只能有 16 个端口，不能再扩展，以此类推。目前这种固定端口的交换机基本上都属较低档次的。模块化交换机上就是交换机上除了有部分固定的端口外，还可通过插入扩展模块，来扩展端口数量、所支持的传输介质 / 网络协议 / 业务类型。图 5-38a、b 所示分别为一款固定端口交换机和一款模块化交换机。

a）固定端口

b）模块化

图 5-38　固定端口交换机和模块化交换机示例

（3）按是否支持网管功能划分

按交换机是否支持网络管理功能可以将交换机划分为网管型和非网管型两大类。网管型交换机可以通过控制端口（Console）或 Web 界面进行配置与管理，非网管型交换机则不能进行任何配置与管理，仅可按照出厂的默认设置进行工作，也就是通常所说的傻瓜型交换机。

从外观上基本上可以判断一个交换机是否具有网管功能，因为网管型交换机都有一个 Console 控制端口，一般为 RS—232 串口型的（也有用普通的 RJ—45 接口的，但会有一个 Console 标注的字样）。图 5-39 所示为一个带有 RS—232 控制端口的网管型交换机。

图 5-39　网管型交换机

5.8.4 二层交换原理

二层交换机与前面介绍的网桥一样，也是工作在 OSI 参考模型的数据链路层，可以直接根据帧中的目的 MAC 地址把数据发送给相应端口上连接的主机。在二层交换机中也有用于数据帧转发的 MAC 地址与端口的映射表（也就是前面所说的 CAM 表），列出了哪个 MAC 地址连接的是哪个端口。当在映射表中没有数据帧中对应的目的 MAC 地址时才进行“泛洪”（以复制方式在除源端口外的其他所有端口上进行转发）。但是要注意，交换机缓存空间毕竟有限，可以存储的 MAC 地址和端口映射表项也限，所以当网络比较大时交换机中的缓存空间就不能保存网络中所有节点 MAC 地址与交换机端口的映射关系了。

1. CAM 表的建立

交换机的这张 CAM 表可以通过多种方式获得，比如静态配置、动态学习，针对多播还可以通过各种多播协议，比如 IGMP 嗅探、GMRP 协议等方式（注意，多播转发表不能通过学习获得，而且多播转发项跟普通转发项不同，跟其对应的出口不止一个，而是一个出口集合，多播方面的具体知识参见笔者编著的《Cisco/H3C 交换机高级配置与管理技术手册》一书）。

在进行数据转发的同时，交换机还有一个学习的过程，它包括两个方面：①交换机在接收到数据帧时会把其中的源 MAC 地址提取出来，查询 CAM 表，看 CAM 表中是否有针对该 MAC 地址的转发项，如果没有，则把该 MAC 地址和接收到该 MAC 地址的端口绑定起来，插入 CAM 表项，这样当接收到一个发送到该 MAC 地址的数据帧时，就不需要向所有端口广播，而仅仅向这一个端口发送即可。②在接收到的数据帧中目的 MAC 地址未知情况下，通过向交换机上其他所有端口进行广播，接收广播帧的节点应答广播帧后，便获知了原来数据帧中对应的目的 MAC 地址所连接的端口，这时交换机又会把该 MAC 地址与所连端口的对应表项插入到 CAM 表中。

注意 数据帧的转发是依据目的 MAC 地址查询 CAM 表，而 CAM 表的学习则是以源 MAC 地址为依据的。但要注意的是，交换机动态学习的 CAM 表项并不是一成不变的，而是启动一个定时器，当该定时器递减到零时，该 CAM 表项被删除。每使用一次该 CAM 表项进行转发，就恢复定时器初始值。

上述介绍是在没有考虑 VLAN（虚拟局域网）的情形下阐述的，现在的交换机一般都实现了 VLAN，所以 CAM 表就有了变化，由原来的两项对应关系（MAC 地址与所连接的交换端口）变成了三项对应关系（MAC 地址，VLAN ID，交换机端口），这样当接收到一个数据帧的时候，交换机就要同时根据数据帧的目的 MAC 地址和 VLAN ID 两项来查询 CAM 表，找到接口后把该数据帧转发出去。

如果交换机根据 MAC 地址和 VLAN ID 查询 CAM 表失败，即没有该 MAC 和 VLAN ID 的对应关系，则交换机把该数据帧对该 VLAN 包含的（除接收端口以外的）所有端口进行广播。但如果只根据 CAM 表来确定一个 VLAN 包含哪些端口，则必须遍历整个 CAM 表，

这样如果 CAM 表的规模非常大，则查询的效率非常低，所以一般的交换机上在实现 VLAN 时，还会创建另外一张表，即 VLAN 配置表。该表包含了 VLAN ID 和交换机上所有端口的对应关系，即只要根据 VLAN ID 查询该表就可以找到该 VLAN 包含的所有端口，这样在进行 VLAN 内广播的时候，就非常容易了。

2. 二层交换原理

因为交换机有多个端口，可直接连接主机或其他交换机。当数据帧发送到本交换机所连接的主机时，交换机就可以根据帧中目的 MAC 地址直接把数据从对应的端口上发送到所连接的主机上。如果数据发送到本交换机所连接的其他交换机上的主机时，则本交换机先把该数据帧发送到连接到对应交换机的端口上，然后再由那台交换机根据目的 MAC 地址从对应端口上发送到目的主机上。

总体来说，二层交换原理与网桥的数据交换原理差不多，具体如图 5-40 所示。不同的只是现在的交换机端口通常不是连接集线器，而是直接交换机和主机，所以在每个端口所连接的物理网段中也采用数据交换方式，而不采用集线器那样的复制类型的广播方式。下面进行具体的解析。

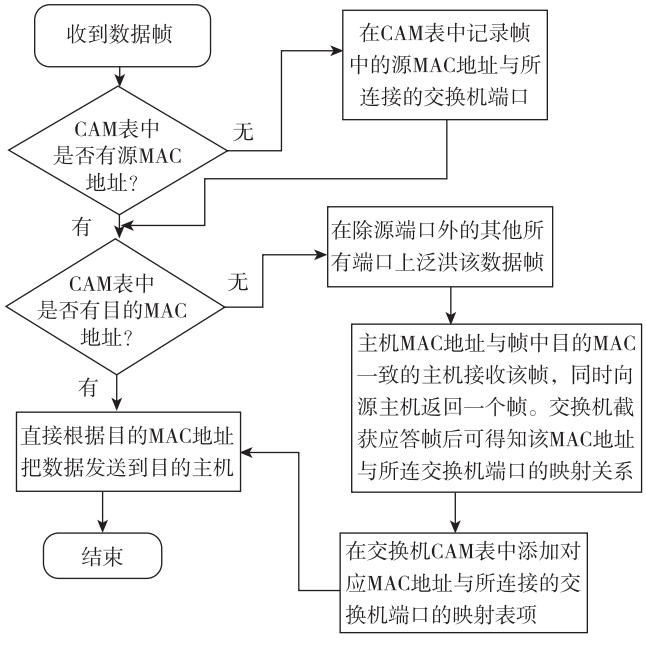


图 5-40　二层交换原理

1）当交换机从某个端口收到一个数据帧后，先读取帧头部的源 MAC 地址，并与自己缓存中的映射表（CAM 表）进行比较，如果没有找到，则在 CAM 表中添加一个该源 MAC 地址与发送该帧的源端口映射表项。这就是交换机的 MAC 地址自动学习功能。

2）如果在 CAM 表项查到了帧中源 MAC 地址，则继续查看是否有帧中目的 MAC 地址所对应的映射表项。如果有，则直接把该帧转发到目的 MAC 地址节点所连接的交换机端口，然后由该端口发送到目的主机。

3）如果在交换机 CAM 表中没有找到帧中目的 MAC 地址所对应的表项，则把该数据帧向除源端口外的其他所有端口上进行泛洪。

4）当 MAC 地址与帧中目的 MAC 地致的主机接收了该数据帧后就会向源主机产生一个应答帧，交换机获取该应答帧后从其中的源 MAC 地址中获取了对应的 MAC 地址和所连接端口的映射关系，并添加到 CAM 表中。这样下次再有 MAC 地址为这个 MAC 地址的帧发送时交换机就可以直接从 CAM 表中找到对应的转发端口，直接转发，不用再泛洪了。

Understanding Computer Networks

王达老师是国内计算机网络领域的权威专家，自2004年以来，撰写了大量关于计算机网络的著作，并多次获得媒体和行业颁发的各种荣誉和奖项，深受读者欢迎。本书是王达老师在该领域的集大成之作，不仅融入了自己多年技术实践经验，而且多年来积累的图书写作心得与技巧也在本书中得到了极致的发挥。它基于最新的网络技术，全面地讲解了计算机网络的基本原理、体系结构和通信协议，及与计算机网络相关的技术的各个方面。强烈推荐！

—— 51CTO 中国领先的IT技术社区（www.51cto.com）

本书主要内容：

- 数制类型及其表示方法、不同数制之间的转换、二进制数的运算及其表示形式。
- 计算机网络的组成和应用、计算机网络的分类、计算机网络的拓扑结构。
- 典型的计算机网络体系结构、计算机网络体系结构通信原理、网络体系结构的设计原理、网络体系结构中的通信协议。
- 物理层的作用、特性、数据通信基础、数据传输速率与信道带宽、数字基带信号编码、信号调制与解调、信道多路复用技术、物理层接口等。
- 数据链路层的功能、结构、实现原理、差错控制方案、流量控制、面向字符的BSC同步传输协议、面向比特的SDLC和HDLC同步传输协议、面向字符的PPP同步传输协议、数据链路层的主要网络设备及其工作原理。
- 介质访问控制子层的作用和原理、CSMA/CD介质访问控制原理、局域网标准以及以太网帧格式、标准以太网规范及体系结构、快速以太网规范及体系结构、千兆以太网规范及体系结构、万兆以太网规范及体系结构、IEEE 802.1D协议、IEEE 802.1Q协议、IEEE 802.1W协议、主要WLAN标准与技术。
- 网络层的作用、数据交换技术、网络层协议及报文格式、路由和路由算法、网络拥塞控制方法和原理。
- IPv4地址的相关技术、IPv4子网划分与聚合、IPv4 NAT基础、 IPv6地址基础及其自动配置；RIP、OSPF、IS-IS、BGP等路由的协议。
- 传输层的作用和原理、传输层的服务功能、TCP协议基础及其可靠性传输、TCP的流量控制和拥塞控制、UDP协议。
- 应用层的作用和原理、Web服务基础、DNS服务原理、DHCP服务原理、电子邮件服务原理。

上架指导：计算机/网络协议
ISBN 978-7-111-41188-8
9 787111 411888 >
定价：89.00元

客服热线：(010) 88378991 88361066
购书热线：(010) 68326294 88379649 68995259
投稿热线：(010) 88379604
读者信箱：hzjsj@hzbook.com
华章网站：www.hzbook.com
网上购书：www.china-pub.com