第 3 章
数据链路层

数据链路层把物理层的原始传输设施转换成一条点到点的通信链路，其主要任务是成帧、寻址、流量控制、差错控制和介质访问控制。它将来自网络层的数据加上头部，封装成帧，并在收、方双方传输速率不一致时进行流量控制。数据链路层利用差错控制技术，对数据进行差错检测和纠错。在传输设备共享传输链路时，它通过数据链路层协议分配通信链路。

本章首先讨论差错控制、数据链路控制和介质访问控制，接着讨论这些技术在局域网中的实现。

本章主要讨论下面 5 个问题：

- 差错控制
- 数据链路控制
- 介质访问控制
- 局域网技术
- 无线局域网

3.1 差错控制

数据在通信系统中传输时，无论通信系统的可靠性有多高，都不可避免地存在干扰，也就是说，任何通信系统都无法避免传输差错的产生。因此，必须在数据传输中考虑使用有效的方法和手段来避免和减少差错，即在数据传输中进行差错控制，包括检错和纠错。

3.1.1 差错的产生原因及类型

在数据传输中，接收端收到的数据与发送端不一致的现象称为差错。出现差错的原因大致分为随机差错和突发性差错两类。

1. 随机差错

随机差错由热噪声引起。通信线路中的热噪声是由传输介质中电子的热运动产生的，克劳德·艾尔伍德·香农关于噪声信道传输速率的结论就是针对这种噪声的。热噪声时刻存在，幅度较小，频谱较宽，是一类随机噪声，一般只会影响个别位，只影响局部范围，在传输时较少发生。在通信线络中，信噪比越高，热噪声引起的差错越少。

2. 突发性差错

突发性差错一般由冲击噪声引起。冲击噪声幅度较大，是一种突发性噪声，由外界环境的电磁干扰引起，例如打雷、闪电时产生的电磁干扰，电焊机引起的电压波动等情况。这种干扰由于持续时间短而幅度大，往往会引起一个位串出错。突发性差错能够影响全局范围，比随机差错更容易发生，所以大部分差错都是这种类型。

3.1.2 检错与纠错

差错控制包括检错和纠错两方面。在数据链路层，差错控制通常指差错检测和重传的方法。

在差错检测中，检错只关心收到的信号是否发生差错，对于差错有多少，差错发生的位置在哪里，都不感兴趣。

在纠错中，需要知道发生差错的位置、差错破坏的位数，还要纠正发生的差错。因此，纠错比检错难度更高。可以推测，要在 10 位数据单元中纠错 2 位，需要考虑 45 种位置；在 100 位中纠正 2 位，则有 4950 种可能；那么，从 1000 位中查 2 位差错呢？由此可见纠错的难度有多高。

检错或纠错的核心就是通过编码技术实现冗余，即为了实现检测或纠正错误，在传输数据时，除了需要传送的数据，加上一些额外的位，接收方通过冗余位实现对传输数据的检错和纠错。冗余位由发送方添加，并由接收方去除，其作用就是实现检错或纠错。

为了提高传输质量，一方面，要尽量提高通信设备的信噪比；另一方面，应采用有效的差错控制方法。常见的差错控制方法有自动请求重传(ARQ)、前向纠错(FEC)和混合纠错(HEC)3 种。

1. 自动重传请求(ARQ)

采用自动重传请求(automatic repeat request，ARQ)方法，发送端将信息码和冗余码组合编成能够进行差错检测的码组发送到信道，接收端收到后进行检验，将检验结果(有误码或者无误码)通过反向信道反馈给发送端，作为对发送端的应答信号。发送端根据收到的确认信息(acknowledgement，ACK)做出是继续发送新的数据，还是重发出错数据的判断。

ARQ 的优点是译码设备简单，对突发错误和信道干扰较严重时有效，因而被广泛地应用在分组交换网络中。其缺点是需要反馈 ACK，实时性差；ARQ 工作时需要接收方发送 ACK，增加了网络负担，也影响了传输速度。采用重复发送数据包来纠正错误的方法，也严重地影响传输速度。

2. 前向纠错(FEC)

前向纠错(forward error correction，FEC)是指信号在被传输之前先对其按一定的格式处理，接收端按规定的算法解码，通过冗余位进行推测，达到找出错码并纠错的目的。

在 FEC 方式中，接收端不但能发现差错，而且能确定发生错误的位置，加以纠正。FEC 发现错误后无须通知发送方重发，接收方能够根据冗余码纠正错误。这一点有别于

ARQ 纠错方式。

FEC 只要求单向信道，因此特别适用于只能提供单向信道的场合，也适用于一点发送、多点接收的广播方式。因为不需要对发送端反馈信息，所以具有接收信号的延时小、实时性好的优点。这种纠错系统的缺点是设备复杂、成本高，且纠错能力越强，编译码设备越复杂。

FEC 技术的产生和发展源于通信系统本身的需求。在工程实践中不存在理想的数字信道，信号在各种媒介的传输过程中总会产生畸变和非等时时延，对数字信号来说，意味着产生误码和抖动，而抖动的最终效果反映在系统的误码上。FEC 编解码可以用硬件实现，也可用软件实现，采用 FEC 技术可较好地改善误码性能。在目前的数字通信系统中，FEC 应用广泛。

3. 混合纠错(HEC)

混合纠错(hybrid error correction，HEC)是前向纠错(FEC)和自动重传请求(ARQ)方式的结合。发送端发送既有自动纠错，又有检错能力的冗余码；接收端收到后，检查差错情况。如果错误在冗余码的纠错能力范围以内，则自动纠错；如果超过了冗余码的纠错能力，但能检测出来，则经过反馈信道请求发送端重发。混合纠错方式在实时性和译码复杂性方面是 FEC 和 ARQ 方式的折中，可达到较低的误码率，较适用于环路延时大的高速数据传输系统。

3.1.3 奇偶校验码

1. 简单奇偶校验码

简单奇偶校验码(simple parity check code)是一种线性块编码。简单奇偶校验是一种最常用的检错方法，只能用于检错，不能纠正任何差错。在这种编码中，n 位数据码变成 m 位数据码，$m=n+1$。其工作原理是：在 n 位数据码后增加 1 位，使码中 1 的个数成为奇数(奇校验)或偶数(偶校验)。增加的位称为奇偶校验位。

【例 3-1】 原始数据＝1101，采用偶校验，则增加校验位后的数据为 11011。

接收方对接收数据进行偶校验，出现错误的情况共有下面 5 种。

(1) 1 位出错。不难看出，数据 11011 的任何 1 位出错，都会破坏码字的偶数性质，从而检测出传输错误。

(2) 2 位出错。出错 2 位时，无法被检测出来。例如，11011 在接收时变成 10111，偶校验位无法检测出错误。

(3) 3 位出错，可以被偶校验位检测出错误。

(4) 4 位出错，不可以被偶校验位检测出错误。

(5) 5 位出错，可以被偶校验位检测出错误。

从上例可以看出，若接收方收到的字节的奇偶结果不正确，可知传输中发生了错误。简单奇偶校验的偶校验只能检测出奇数个比特位错，对偶数个比特位错无能为力。

2. 二维奇偶校验码

简单奇偶校验码只能检测奇数个差错，无法检测出偶数个差错。更好的方法是二维

奇偶校验(two dimensional parity check)。二维奇偶校验数据以表格形式组织,每行和每列都计算出奇偶校验位,然后将整个表发送给接收方;接收方分别对每行和每列进行校验,如图 3-1 所示。

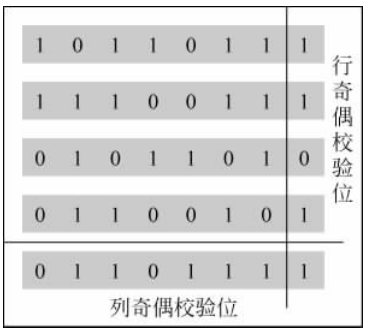

图 3-1 二维奇偶校验码的实现

二维奇偶校验码的检错能力比简单奇偶校验码要强。对于简单奇偶校验码,当出错位数为偶数时,错误无法被检出。二维奇偶校验码出错 1 位、2 位或 3 位时,都会影响到至少 1 个校验位,差错都能被检出。当然,当出错的位数为 4 的倍数时,同样可能出现差错无法被检测出来的情况,如图 3-2 所示。

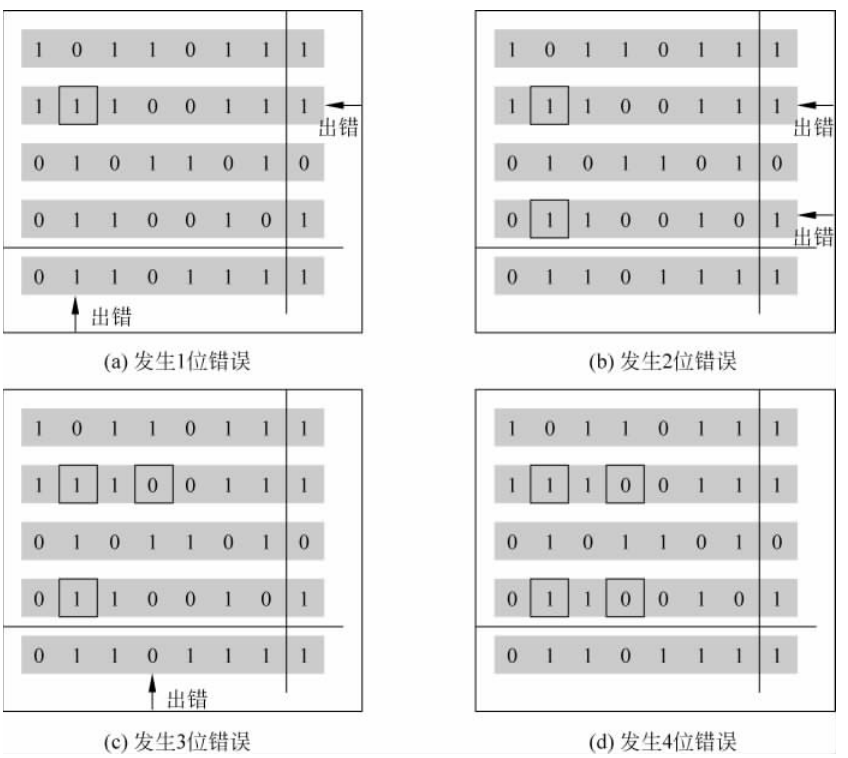

图 3-2 二维奇偶校验码差错检测

3.1.4 海明码

1950 年,海明(Hamming)研究了用冗余数据位来检测和纠正代码差错的理论和方法。按照海明的理论,可以在数据代码上添加若干冗余位,组成码字。码字之间的海明距离(指两个长度相同的字对应位不同的数量)是一个码字要变成另一个码字时必须改变的最小位数。例如,7 位 ASCII 码增加 1 位奇偶位,成为 8 位码字,这 128 个 8 位码字之间的海明距离是 2。所以,其中 1 位出错,便能检测出来;2 位出错时,变成了另外一个码字,错误无法被检出。

海明用数学分析的方法说明了海明距离的几何意义：n 位码字可以用 n 维空间超立方体的一个顶点来表示。两个码字之间的海明距离就是超立方体的两个对应顶点之间的一条边，而且这是两个顶点之间的最短距离，出错的位数小于这个距离，都可以被判断为就近的码字。这就是海明码纠错的原理，它通过增加冗余码位来换取正确率的提高。

按照海明的理论，纠错码的编码就是把所有合法的码字尽量安排在 n 维超立方体的顶点上，使得任意一对码字之间的距离尽可能大。如果任意两个码字之间的海明距离是 d，则所有少于等于 $d-1$ 位的错误都可以检测出来，所有少于 $d/2$ 位的错误都可以纠正。一个自然的推论是：对于某种长度的错误串，要纠正它，就要用比仅仅检测它多 1 位的冗余位。

如果对 m 位数据增加 k 位冗余位，可组成 $n=m+k$ 位纠错码。对于 2^m 个有效码字中的每一个，都有 n 个无效但可以纠错的码字。这些可纠错的码字与有效码字的距离是 1，含单个错误位。这样，对于一个有效的消息，总共有 $n+1$ 个可识别的码字。这些码字与其他 2^m-1 个有效消息的距离都大于 1，意味着总共有 $2^m(n+1)$ 个有效的或是可纠错的码字。显然，这个数应小于等于码字的所有可能的个数，即 2^n，于是有

$$2^m(n+1) < 2^n$$

因为 $n=m+k$，可得

$$M+k+1 < 2^k$$

对于给定的数据位 m，上式给出了 k 的下界，即要纠正单个错误，k 必须取的最小值。海明建议了一种方案，可以达到这个下界，并能直接指出错在哪一位。首先，把码字的位从 1 到 n 编号，并把这个编号表示成二进制数，即 2 的幂之和；然后，对 2 的每一个幂设置一个奇偶位。例如，对于 6 号位，由于 6=110B，所以 6 号位参加第 2 位和第 4 位的奇偶校验，而不参加第 1 位的奇偶校验。类似地，9 号位参加第 1 位和第 8 位的校验，而不参加第 2 位和第 4 位的校验。海明把奇偶校验分配在 1、2、4、8 等位置上，其他位放置数据。下面根据图 3-3 举例说明编码的方法。

		校验位			
		8	4	2	1
数据位	3	0	0	1	1
	5	0	1	0	1
	6	0	1	1	0
	7	0	1	1	1
	9	1	0	0	1
	10	1	0	1	0
	11	1	0	1	1

图 3-3　海明编码例图

假设传送的信息为 1001011，把各个数据放在 3、5、6、7、9、10、11 等位置上，1、2、4、8 位留作校验位。

		1		0	0	1		0	1	1
1	2	3	4	5	6	7	8	9	10	11

根据图 3-3，第 3、5、7、9、11 位的二进制编码的第 1 位为 1，所以 3、5、7、9、11 号位参加第 1 位校验。若按偶校验计算，1 号位应为 1，即

1		1		0	0	1		0	1	1
1	2	3	4	5	6	7	8	9	10	11

类似地，3、6、7、10、11 号位参加 2 位校验，5、6、7 号位参加 4 位校验，9、10、11 号位参加 8 位校验，全部按偶校验计算，最终得到

1	0	1	1	0	0	1	0	0	1	1
1	2	3	4	5	6	7	8	9	10	11

如果这个码字在传输中出错，比如 6 号位出错，即变成

√	×		×				√			
1	0	1	1	0	1	1	0	0	1	1
1	2	3	4	5	6	7	8	9	10	11

当接收端按照同样的规则计算奇偶位时，发现 1 号位和 8 号位的奇偶性正确，2 号位和 4 号位的奇偶性不对，于是 2+4=6，立即可确认错在 6 号位。

在上述例题中，$k=4$，因而 $m<2^4-4-1=11$(位)，共组成 15 位码字，可检测出单个位的错误。

3.1.5 循环冗余校验码

1. 检错原理

循环冗余码 CRC 在发送端编码和接收端校验时，把发送二进制码看作一个多项式 $F(x)$的系数，收、发双方约定一个生成多项式 $G(x)$，其系数对应另一个二进制码(设为 K 位)。先把 $F(x)$对应的二进制码左移 $K-1$ 位，然后用它除以 $G(x)$系数对应的二进制码，得到余数二进制码($K-1$ 位)，最后由要发送码加余数码，构成 CRC 码。接收方收到后，用 $G(x)$除多项式。若有余数，则传输有错，并可根据余数判断错误位置和纠错。CRC 具有很强的检错能力，而且容易用硬件实现，在局域网中应用广泛。

根据应用环境与习惯的不同，CRC 又分为以下几种：CRC-12 码、CRC-16 码、CRC-CCITT 码和 CRC-32 码。

CRC-12 码通常用来传送 6bit 字符串；CRC-16 及 CRC-CCITT 码用来传送 8bit 字符串，其中美国采用 CRC-16，欧洲国家采用 CRC-CCITT；CRC-32 码大都用在一种称为 Point-to-Point(点到点)的同步传输中。

为了能对不同场合下的各种错误模式进行校验，人们研究出几种 CRC 生成多项式的国际标准，如下所示：

CRC-12　　$G(x)=x^{12}+x^{11}+x^3+x^2+x+1$

CRC-16　　$G(x)=x^{16}+x^{15}+x^2+1$

CRC-CCITT　$G(x)=x^{16}+x^{12}+x^5+1$

CRC-32　　$G(x)=x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^3+x^2+x+1$

2. CRC 码计算方法(模 2 除法)

用模 2 除法(又称异或，即二进制位运算。值相同，结果为 0；值不同，结果为 1)计算校验码。

假设 $G(x)$为 n 阶多项式，在帧的低位端加上 n 个 0 位，该帧变成 $m+n$ 位，对应多项式 $x^n \cdot M(x)$。利用模 2 除法，用对应的 $G(x)$去除对应于 $x^n \cdot M(x)$的位串，得到的余数就是校验码。

【例 3-2】 已知数据 $M=1100$，多项式 $G(x)=x^3+x+1$。求 CRC 码。

解：$G(x)$为 3 阶多项式，$n=3$，在数据帧后添加 3 个 0，形成新的数据串 1100000。采用模 2 取余法，计算过程如图 3-4 所示。

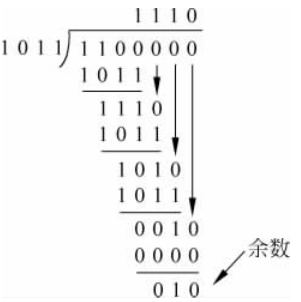

图 3-4 CRC 码计算过程

余数为 010，即 CRC 校验码，传输的帧为 1100010。验证结果如表 3-1 所示。

表 3-1 验证结果

结果	A1	A2	A3	A4	A5	A6	A7	余	数		出错位
正确	1	1	0	0	0	1	0	0	0	0	无
错误							1	0	0	1	7
						0		0	1	0	6
					1			1	0	0	5
				1				0	1	1	4
			1					1	1	0	3
		0						1	1	1	2
	0							1	0	1	1

3.2 数据链路控制

数据链路层的两个主要功能是数据链路控制(data link control，DLC)和介质访问控制(medium access control，MAC)。数据链路控制的功能是确保节点之间正确交换数据，主要包括成帧、流量控制、差错控制以及节点间可靠传输的协议等。介质访问控制主要解决传输介质在多路访问时的控制与协调问题。本节主要介绍数据链路控制。

3.2.1 成帧

在物理层进行数据传输，是指数据以信号的形式从发送端传输到接收端。物理层要提供位同步来保证发送端和接收端的周期和时序相同。数据到达数据链路层时，需将二

进制位组合成帧。帧通过添加发送端和接收端地址，将分组分离。接收端地址标明分组传输的目的地，发送端地址用于接收方确认接收。

数据帧的大小可以是固定的，也可以是可变的，取决于采用何种通信技术。例如，ATM广域网的帧（信元）使用固定长度帧；在局域网中，主要采用可变长度帧。可变长度帧需要用某种方法来规定一个帧的结束和下一个帧的开始，常用的有面向字符的可变长度帧和面向位的可变长度帧。

1. 面向字符协议

在面向字符协议（character oriented protocol）可变长度帧中，数据用类似于 ASCII 编码系统的 8 位字符传输，传送的是由若干字符组成的数据块。这种类型的帧由标记、头部、由字符组成的数据块、尾部等组成，如图 3-5 所示。头部和尾部用于标记帧的开始和结束。头部由发送端地址、接收端地址以及控制信息组成，尾部是检错或纠错冗余码。

图 3-5 面向字符协议可变长度帧

面向字符成帧适合传输文本时使用，标记可以选用通信中不使用的任意字符。但在传输图片、音频、视频时，可能遇到标志帧以及开始和结束的帧标记成为信息的一部分，出现帧边界难以区分的问题。为此采用字符填充策略，如果数据中出现与标记相同的字符，则在帧的数据部分填入一个特殊的字节充当转义字符，当接收方遇到转义字符时，自动将转义字符去除，并将下一字符按数据处理，而不当成是分界标记。

2. 面向比特协议

面向比特协议（bit oriented protocol）又称面向位协议。在面向比特协议可变长度帧中，帧由标记、头部、由位组成的数据块、尾部等组成，如图 3-6 所示。多数协议使用 8 位模式 01111110 作为帧的开始和结束的标记。

图 3-6 面向比特协议可变长度帧

面向比特协议标记也会出现与面向字节协议标记相同的问题，即当标记出现在数据中时，需要告知接收端这并不是帧的结束标记。在面向比特协议中，数据通过位填充方法来区别标记。此时，若遇到 011111，不管后一位是什么，都增加一个 0（接收方接收时去除），使接收方不会误以为这是一个标记。换言之，如果有标记模式 01111110 出现在数据中，位填充将使数据变为 011111010，接收方不会误认为它是标记。

3.2.2 流量控制和差错控制

数据链路控制最重要的功能就是流量控制和差错控制，确保各节点之间正确地收发数据。

1. 流量控制

流量控制是数据链路层的一项重要功能，它在接收方确认前协调发送方发送的数据数量，保证收、发双方协调工作。数据处理过程通常比传输速度慢，因此，每台接收设备都设有一块存储区，称为缓冲区，存储接收的数据，直到它们被处理，进入缓冲区的数据必须经过校验和处理才能使用。

在大多数协议中，流量控制是一系列程序，用于告知发送方在等待来自接收方的确认之前，它能传输多少数据。任何接收设备都有对于进入缓存的数据的处理速度的限制，以及进入缓存的数据的存储容量限制。接收设备必须能够在达到这些限制前提示发送设备，并要求发送端减少或暂停数据帧的发送。

2. 差错控制

差错控制包括检错和纠错，以便接收方识别传输中的错误，并告知发送方重传丢失或损坏的数据帧。

数据链路层的差错控制主要指差错检测和重传的方法。在数据链路层，差错控制实现起来较为简单，接收方只要检测到差错，就会要求发送方重传出错的数据帧。这个过程称为自动重传请求(automatic repeat request，ARQ)。

3.2.3 数据链路控制

数据链路层通过数据链路控制协议实现成帧、流量控制和差错控制，保证节点之间正确交换数据帧。

1. 无噪声通道

完全理想化的数据传输基于两个假定：假定一，通信链路无噪声，数据在传输时不存在任何差错，也不会丢失帧、复制帧和损坏帧。假定二，接收端处理数据的速率不低于发送端发送数据的速率。

1) 最简单协议

基于无噪声通道的两个假定，在最简单协议中，不会发生差错控制，也不需要流量控制。发送端数据链路层从网络层获取数据后，将数据组装成帧，传到物理层发送。接收端从物理层接到帧后，从帧中提取出数据，送往网络层。数据链路层为网络层提供传输服务，并使用物理层提供的服务进行位的传输。在数据帧传输中，发送方发送一个帧的序列完全不用考虑接收方，如图 3-7 所示。

2) 停止等待协议

如果上述两个假定中，第一个假定存在，第二个假定不存在，即接收方无法保证处理数据的速率不低于发送方发送数据的速率，就必须考虑流量控制问题。如果到达接收方的数据大于能被处理的数据，帧在处理前必须先存储。当接收方存储空间不足时，接收到

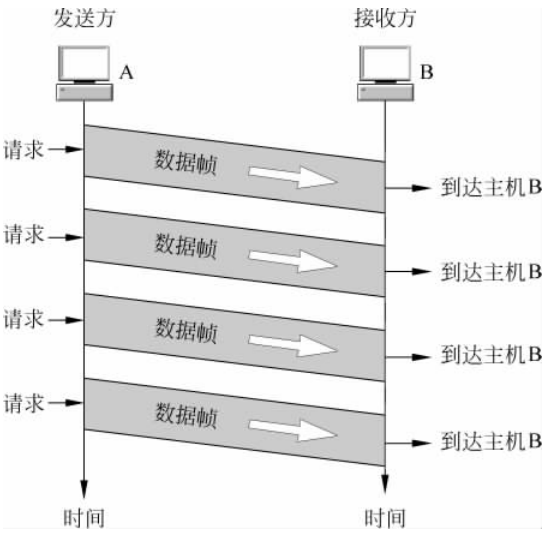

图 3-7 无流量控制的帧发送

的数据帧将被丢弃或拒绝。为了避免接收端出现帧超负荷，必须给发送端发送信息，通知它减缓数据帧发送速度。

在停止等待协议中，发送端发出一个数据帧后，必须要在收到接收端的确认帧(acknowledgment，ACK)之后，才可以发送下一个数据帧，如图 3-8 所示。

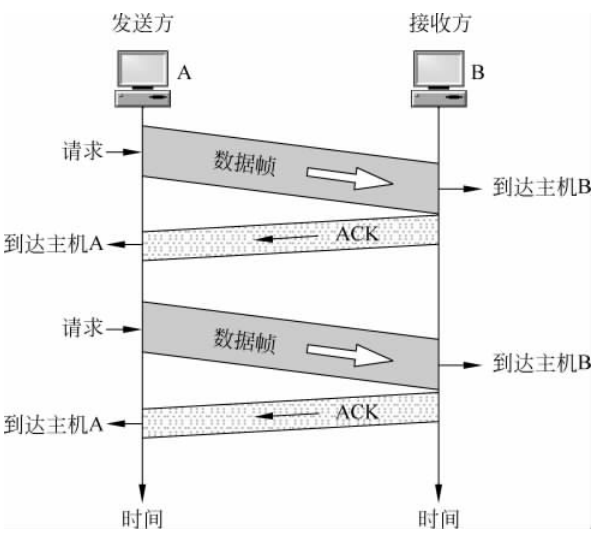

图 3-8 停止等待协议

2. 有噪声通道

无噪声通道中的两个假设在现实中是不存在的。在现实的数据帧传输中，不但要进行流量控制，也要进行差错控制。