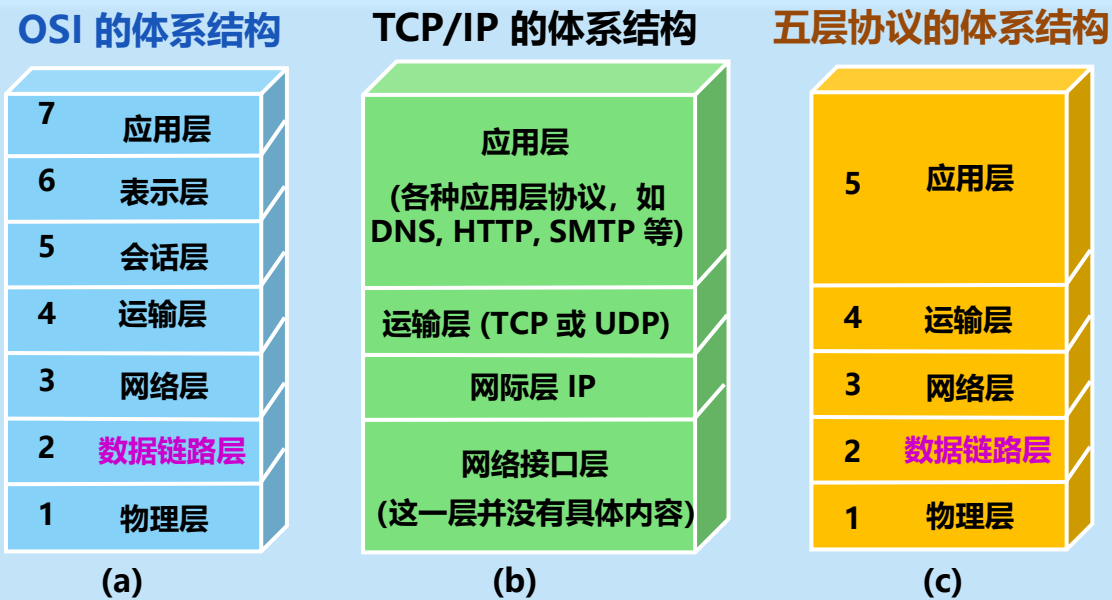
计算机网络（第5版）

Andrew S. Tanenbaum 编著
清华大学出版社

第 3 章

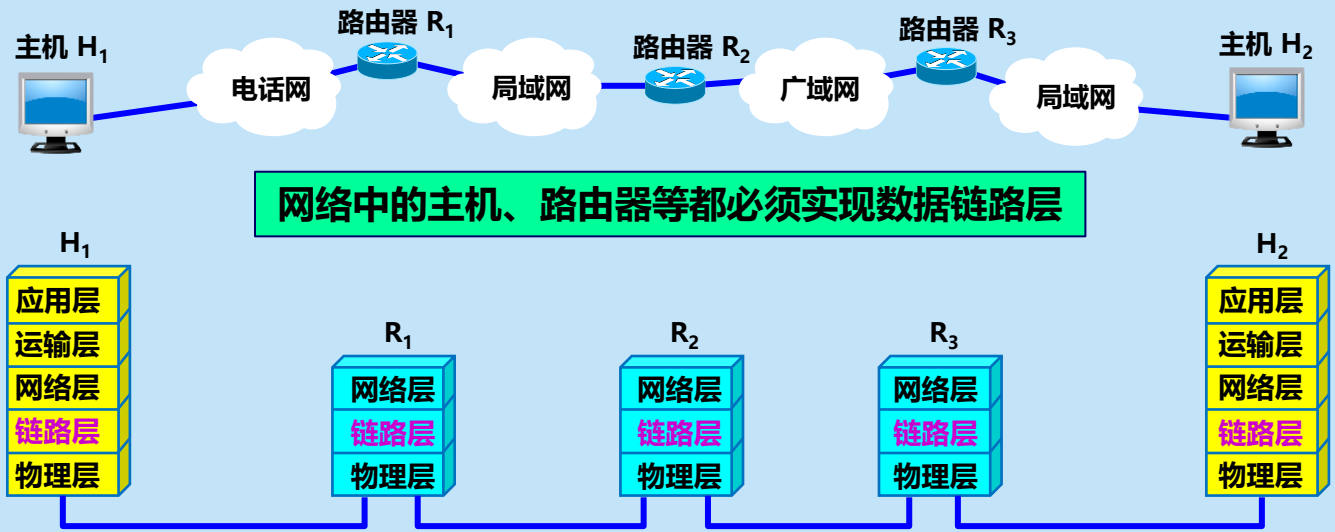
数据链路层

计算机网络体系结构

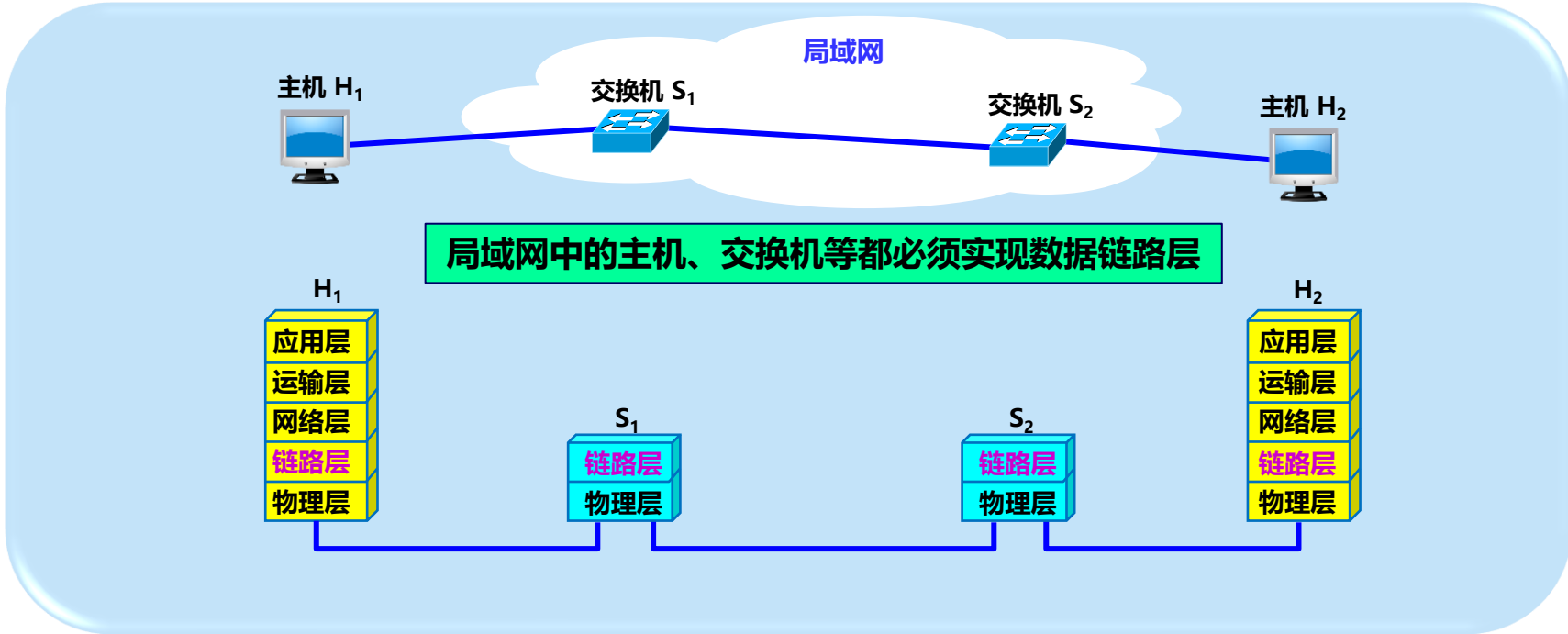


计算机网络体系结构：(a) OSI 的七层协议；(b) TCP/IP 的四层协议；(c) 五层协议

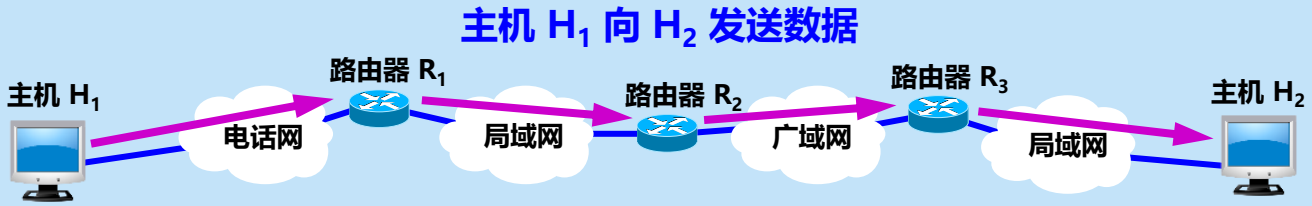
数据链路层是实现设备之间通信的非常重要的一层



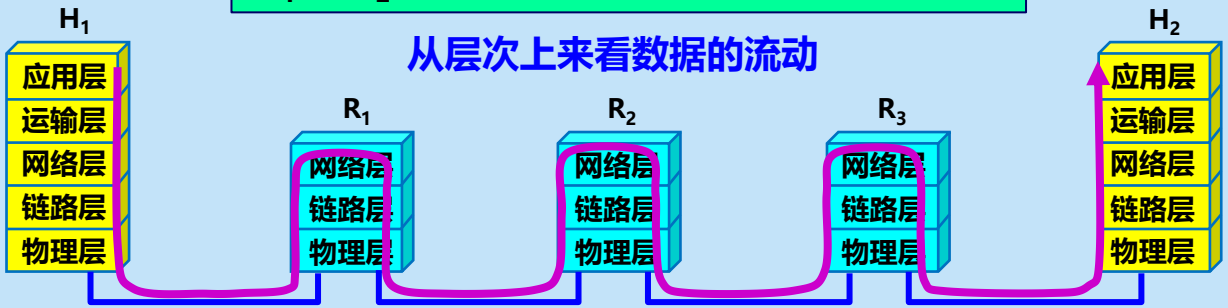
数据链路层是实现设备之间通信的非常重要的一层



数据链路层的作用



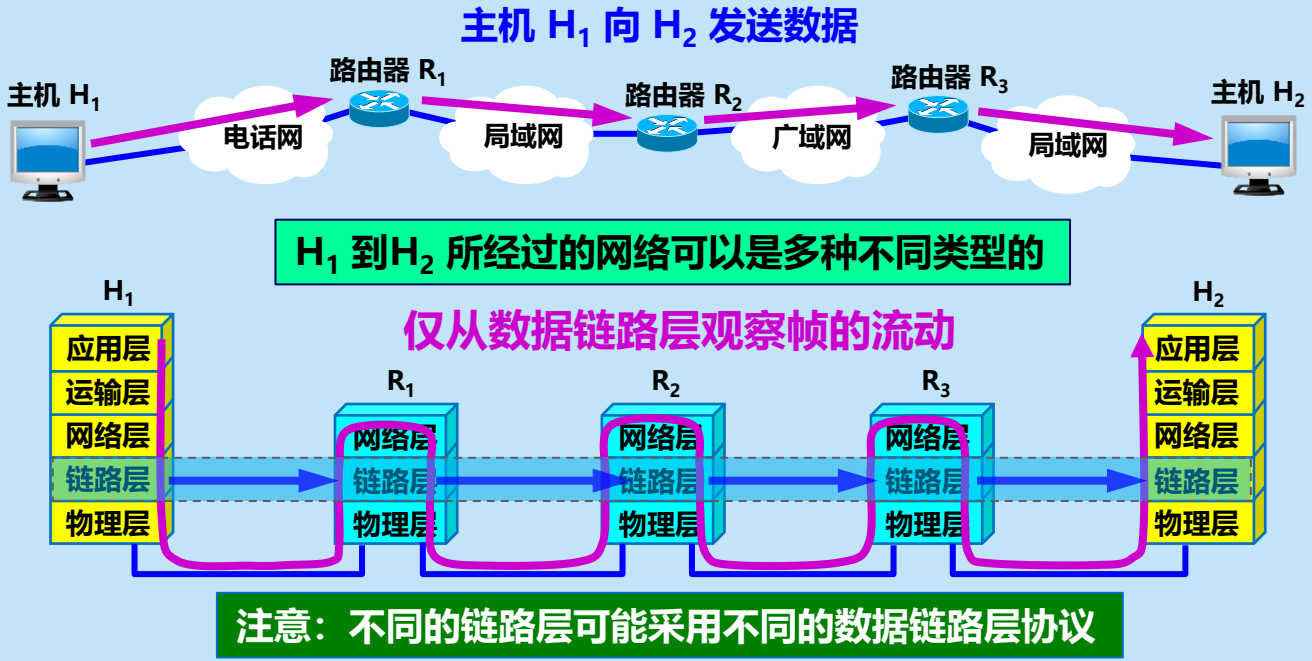
H_1 到H_2 所经过的网络可以是多种不同类型的



从层次上来看数据的流动

数据链路层的地位

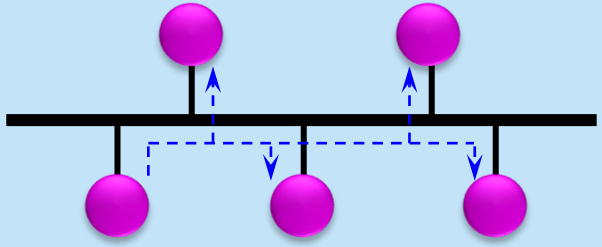
数据链路层的作用


主机 H1 向 H2 发送数据
主机 H1
路由器 R1
路由器 R2
路由器 R3
主机 H2
电话网
局域网
广域网
局域网
H1 到H2 所经过的网络可以是多种不同类型的
H1
H2
仅从数据链路层观察帧的流动
R1
R2
R3
应用层
运输层
网络层
链路层
物理层
注意：不同的链路层可能采用不同的数据链路层协议

数据链路层使用的信道

(a) 点对点信道

- 这种信道使用一对一的**点对点**通信方式。



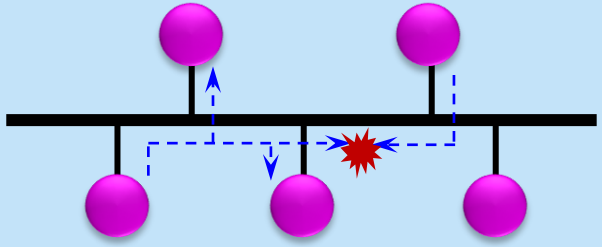
(b) 广播信道

- 使用一对多的**广播通信**方式。
- 因此必须使用专用的共享信道协议来协调这些主机的数据发送。

数据链路层使用的信道

(a) 点对点信道

- 这种信道使用一对一的**点对点**通信方式。



(b) 广播信道

- 使用一对多的**广播通信**方式。
- 因此必须使用专用的共享信道协议来协调这些主机的数据发送。

3.1 数据链路层的设计问题

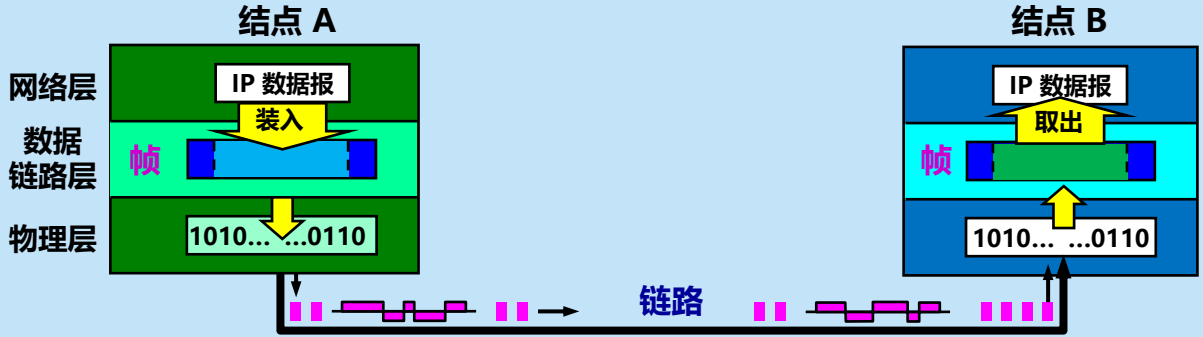
3.1.1 数据链路和帧

- 链路 (link) 是一条无源的点到点的物理线路段，中间没有任何其他的交换结点。
  - 一条链路只是一条通路的一个组成部分。
- 数据链路 (data link) 除了物理线路外，还必须有通信协议来控制这些数据的传输。若把实现这些协议的硬件和软件加到链路上，就构成了数据链路。
  1. 现在最常用的方法是使用适配器（即网卡）来实现这些协议的硬件和软件。
  2. 一般的适配器都包括了数据链路层和物理层这两层的功能。

3.1.1 数据链路和帧

- 也有人采用另外的术语。这就是把链路分为物理链路和逻辑链路。
- **物理链路**就是上面所说的链路。
- **逻辑链路**就是上面的数据链路，是物理链路加上必要的通信协议。
- 早期的数据通信协议曾叫做**通信规程** (procedure)。因此在数据链路层，规程和协议是同义语。

数据链路层传送的是帧



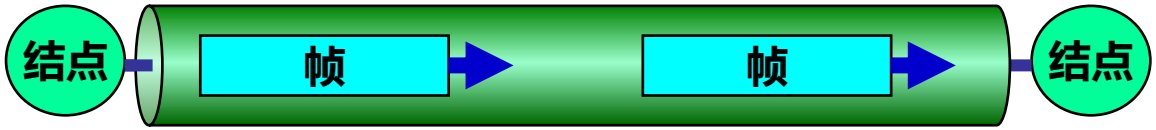
(a) 三层的简化模型

(b) 只考虑数据链路层

使用点对点信道的数据链路层

数据链路层像个数字管道

- 常常在两个对等的数据链路层之间画出一个数字管道，而在这条数字管道上传输的数据单位是帧。



- 数据链路层不必考虑物理层如何实现比特传输的细节。甚至还可以更简单地设想好像是沿着两个数据链路层之间的水平方向把帧直接发送到对方。

提供给网络层的服务

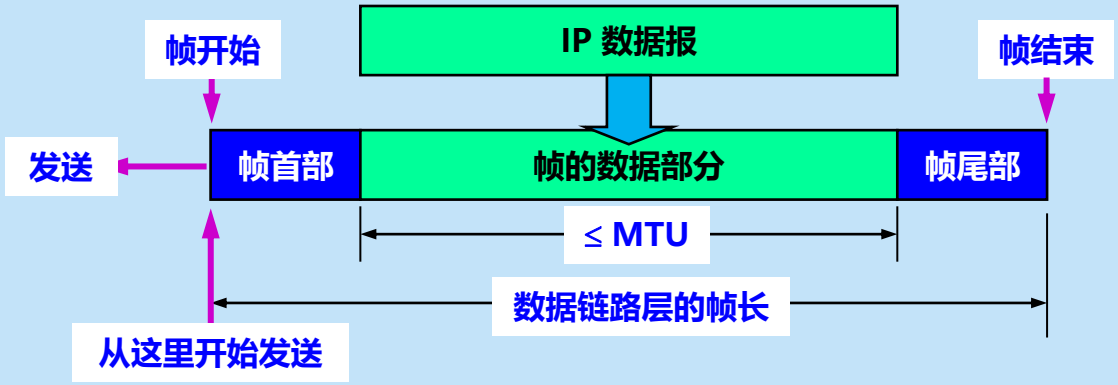
- 数据链路层可以设计成向上提供各种不同的服务。实际提供的服务因具体协议的不同而有所差异。一般情况下，数据链路层通常会提供以下三种可能的服务：
  1. 无确认的无连接服务
  2. 有确认的无连接服务
  3. 有确认的有连接服务

3.1.2 三个基本问题

- 数据链路层协议有许多种，但有三个基本问题则是共同的。这三个基本问题是：
  1. 封装成帧 (开始标志和结束标志)
  2. 透明传输 (转义字符的处理)
  3. 差错控制 (对帧的检错)

1. 封装成帧

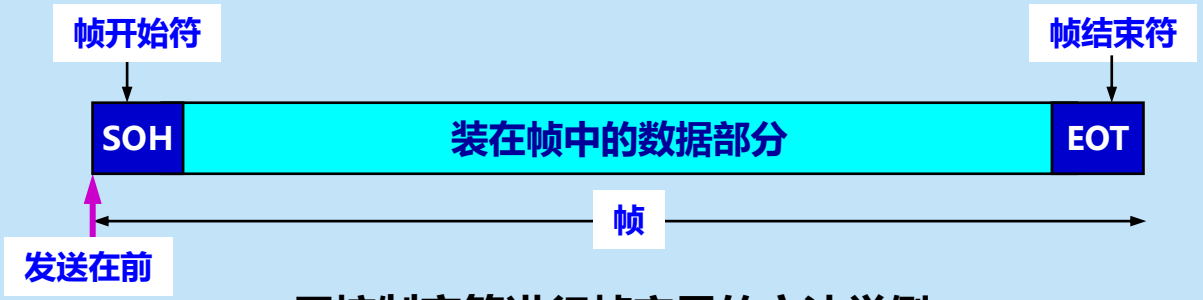
- **封装成帧** (framing) 就是在一段数据的前后分别添加**首部和尾部**，然后就构成了一个帧。
- 首部和尾部的一个重要作用就是进行**帧定界**。



用帧首部和帧尾部封装成帧

用控制字符进行帧定界的方法举例

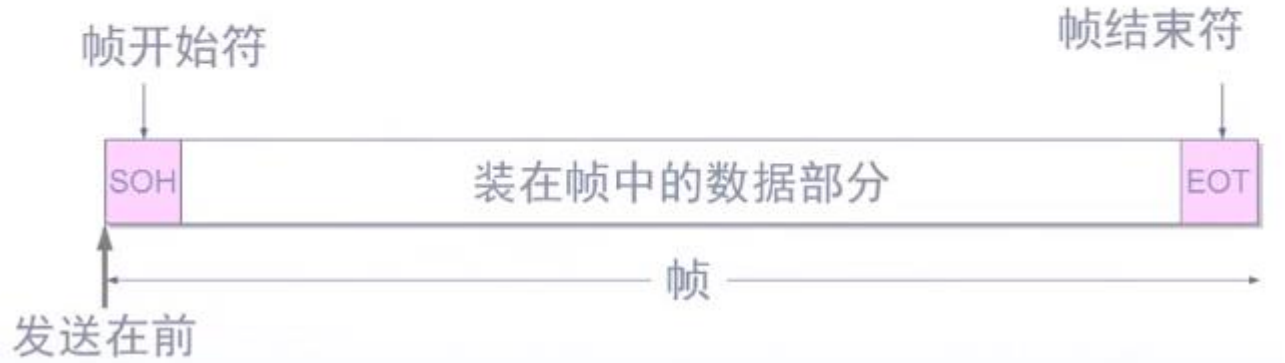
- 当数据是由可打印的 ASCII 码组成的文本文件时，帧定界可以使用特殊的帧定界符。（可打印的 ASCII字符：计算机键盘按下去能够在屏幕上显示的字符）
- 控制字符 SOH (Start Of Header) 放在一帧的最前面，表示帧的首部开始。另一个控制字符 EOT (End Of Transmission) 表示帧的结束。



用控制字符进行帧定界的方法举例

用控制字符进行帧定界的方法举例

试想：帧还未发送完，发送端出了问题，只能重发该帧。接收端丢面的“半截子帧”，它会抛弃吗？为什么？



为什么给帧同时添加首尾标识符？

若该帧只添加了SOH

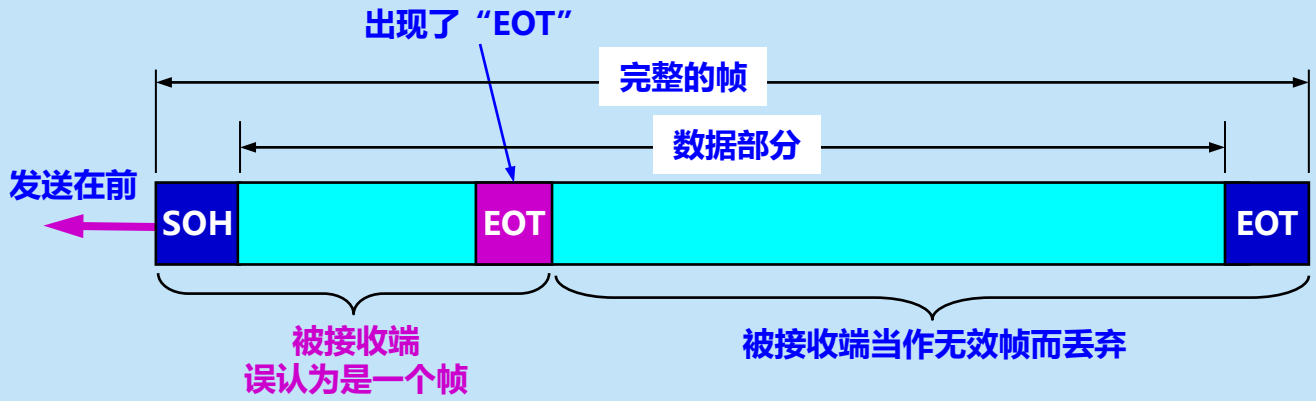
当PC发送数据，接收端没有收到EOT，该帧为“半截子帧”，接收端会丢弃该帧

反之亦然

PC只有同时收到SOH和EOT，才会认为这个帧是完整的

2. 透明传输

- 如果数据中的某个字节的二进制代码恰好和 SOH 或 EOT 一样，数据链路层就会错误地“找到帧的边界”。



数据部分恰好出现与 EOT 一样的代码

解决透明传输问题

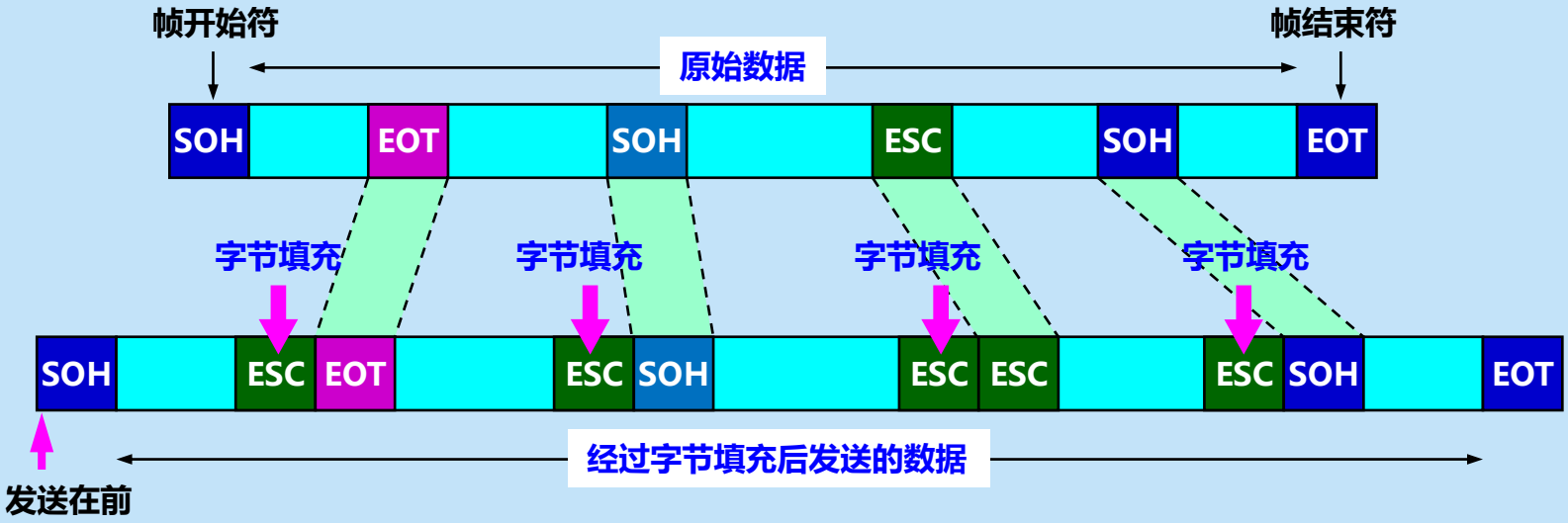
- 解决方法：字节填充 (byte stuffing) 或字符填充 (character stuffing)。
- 发送端的数据链路层在数据中出现控制字符“SOH”或“EOT”的前面插入一个转义字符“ESC”(其十六进制编码是1B)。
- 接收端的数据链路层在将数据送往网络层之前删除插入的转义字符。
- 如果转义字符也出现在数据当中，那么应在转义字符前面插入一个转义字符 ESC。当接收端收到连续的两个转义字符时，就删除其中前面的一个。

透明

- 指某一个实际存在的事物看起来却好像不存在一样。

“在数据链路层透明传送数据”表示无论发送什么样的比特组合的数据，这些数据都能够按照原样没有差错地通过这个数据链路层。

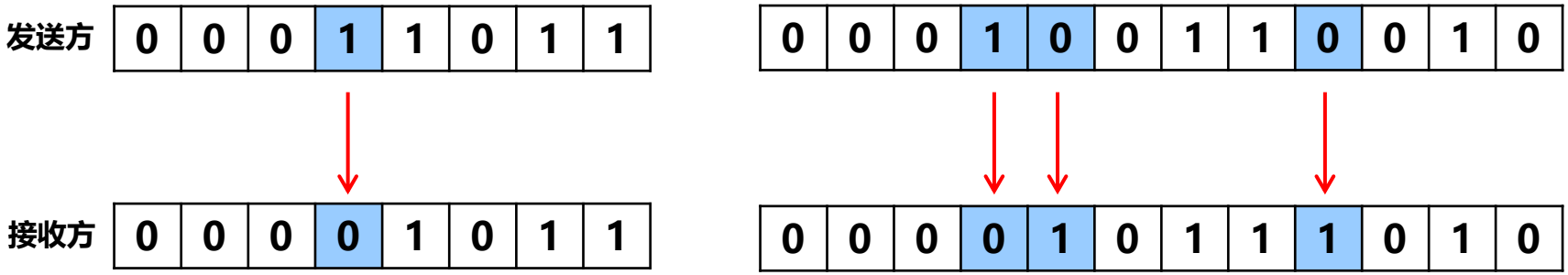
用“字节填充”法解决透明传输的问题



接收端收到后，会自动将多余的ESC去掉

3. 差错检测

在传输过程中可能会产生比特差错：1 可能会变成 0， 而 0 也可能变成 1。



一位比特错　　多位比特错

3．差错检测

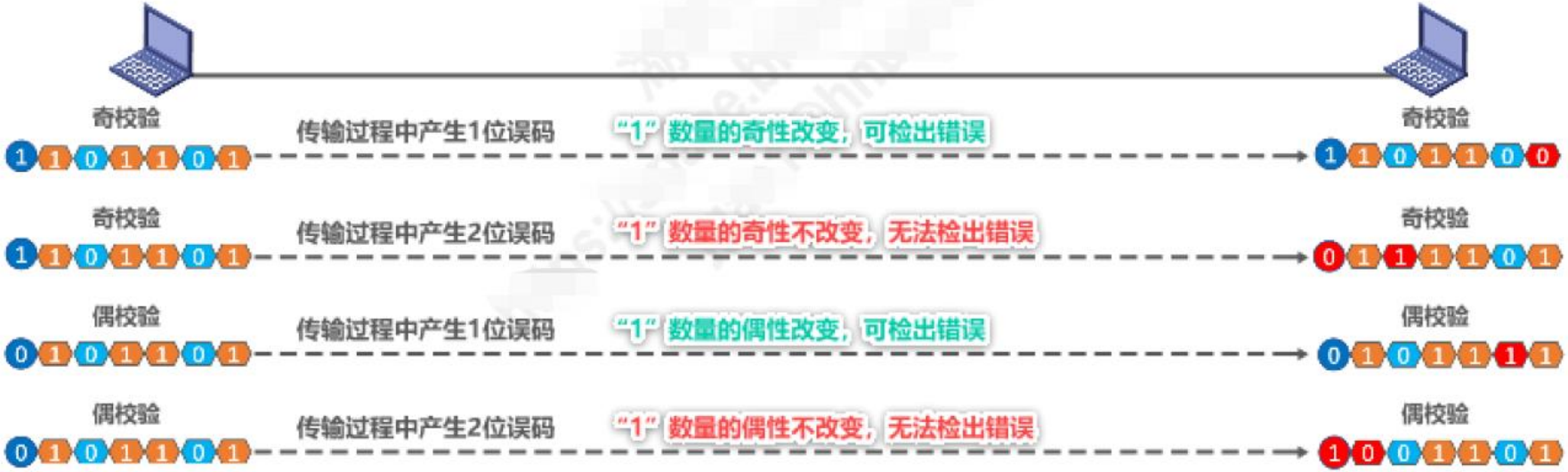
- 在一段时间内，传输错误的比特占所传输比特总数的比率称为误码率 BER (Bit Error Rate)。
- 误码率与信噪比有很大的关系。
- 为了保证数据传输的可靠性，在计算机网络传输数据时，必须采用各种差错检测措施。

以太网V2的MAC帧（最大长度为1518字节）				
6字节	6字节	2字节	46～1500 字节	4字节
目的地址	源地址	类型	数 据 载 荷	FCS

PPP帧的格式						
1字节	1字节	1字节	2字节	不超过1500字节	2字节	1字节
标志	地址	控制	协议	数 据 载 荷	FCS	标志

3.1 奇偶校验

- 在待发送的数据后面**添加1位奇偶校验位**，使整个数据（包括所添加的校验位在内）中“1”的个数为奇数（奇校验）或偶数（偶校验）。
- 如果有**奇数个位发生误码**，则奇偶性发生变化，**可以检查出误码**；
- 如果有**偶数个位发生误码**，则奇偶性不发生变化，**不能检查出误码（漏检）**；



3.1 奇偶校验

特点：

奇偶校验只能检错，不能纠错

若出现两位错，奇偶不变，也无法判断出错

应用：

常用于外部设备

如：键盘输入时使用ASCII码，再配一位校验位，

组成8位的奇偶校验位，正好占一个字节

改进的奇偶校验

- 对数据位组成一个L位宽，K位高的长方形距阵来发送，然后对每一列单独计算奇偶位，并附在最后一行作为冗余位。

差错检测能力分析：

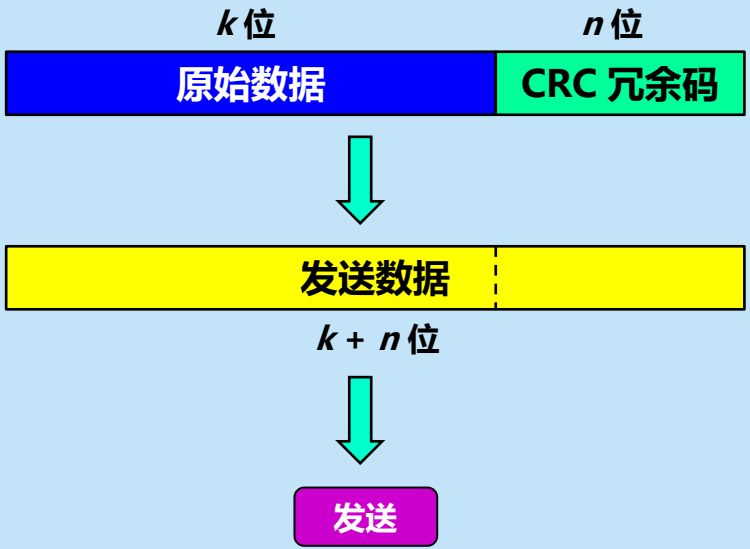
1.该方法可以检测所有长度为L的突发性错误。

2.当出现错误长度大于L时，假设L列中任意一列检测出错的概率为1/2，那么，整个数据块的错判率（实际出错却判为正确的概率）为（1/2）^{L}。

该方法用在ICMP报头检验中。

3.2 循环冗余检验(CRC)的原理

- 在数据链路层传送的帧中，广泛使用了循环冗余检验 CRC 的检错技术。



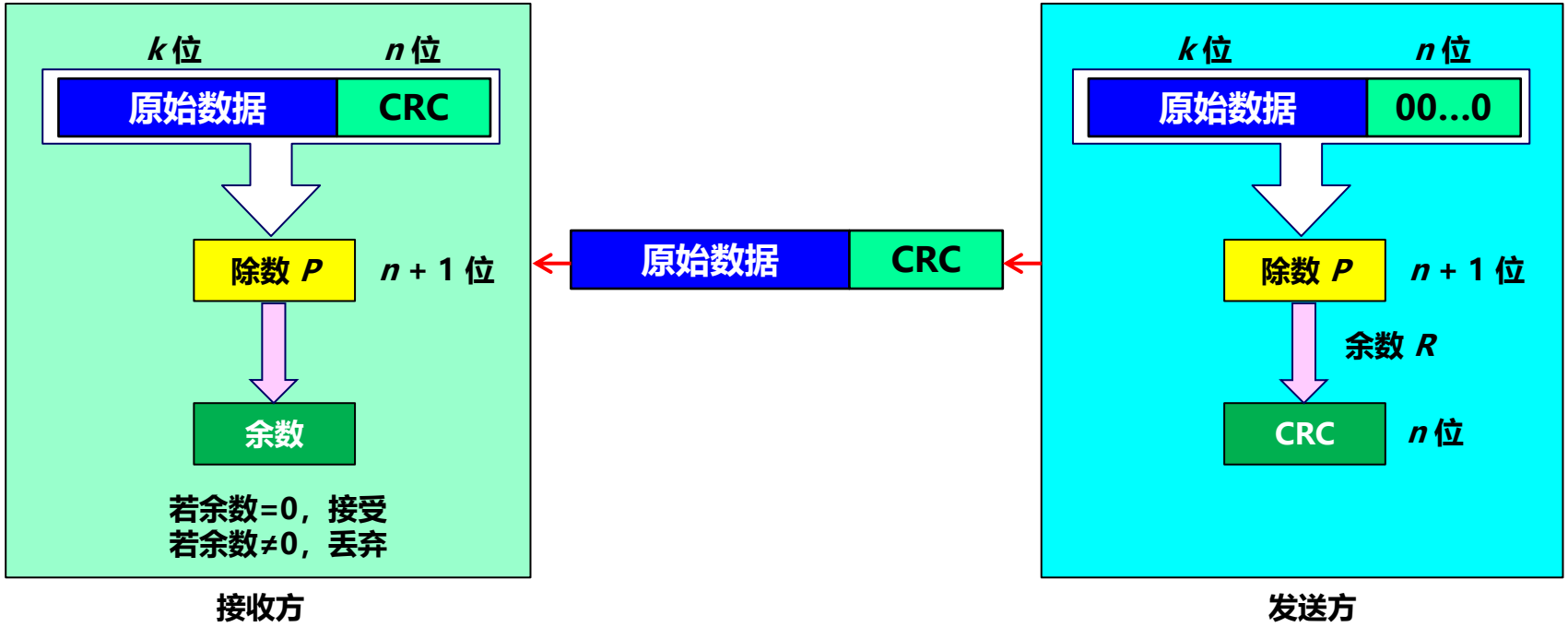
- 在发送端，先把数据划分为组。假定每组 *k* 个比特。
- 在每组 *M* 后面再添加供差错检测用的 *n* 位冗余码，然后一起发送出去。

冗余码的计算

- 用二进制的模 2 运算进行 2^n 乘 M 的运算，这相当于在 M 后面添加 n 个 0。
- 得到的 $(k + n)$ 位的数除以事先选定好的长度为 $(n + 1)$ 位的除数 P，得出商是 Q 而余数是 R，余数 R 比除数 P 少 1 位，即 R 是 n 位。
- 将余数 R 作为冗余码拼接在数据 M 后面，一起发送出去。

- 关于除数P的选定：
- 由数据链路层的协议实现，用户无需关心

冗余码的计算


k 位
n 位
原始数据
CRC
除数 P
n + 1 位
余数
若余数=0，接受
若余数≠0，丢弃
接收方
原始数据
CRC
k 位
n 位
原始数据
00…0
除数 P
n + 1 位
余数 R
CRC
n 位
发送方

接收端对收到的每一帧进行 CRC 检验

- (1) 若得出的余数 $R = 0$，则判定这个帧没有差错，就接受 (accept)。
- (2) 若余数 $R \neq 0$，则判定这个帧有差错，就丢弃。
- 但这种检测方法并不能确定究竟是哪一个或哪几个比特出现了差错。
- 只要经过严格的挑选，并使用位数足够多的除数 P，那么出现检测不到的差错的概率就很小很小。

- 在数据链路层，检测到有错误的帧，直接丢弃，并不进行纠错

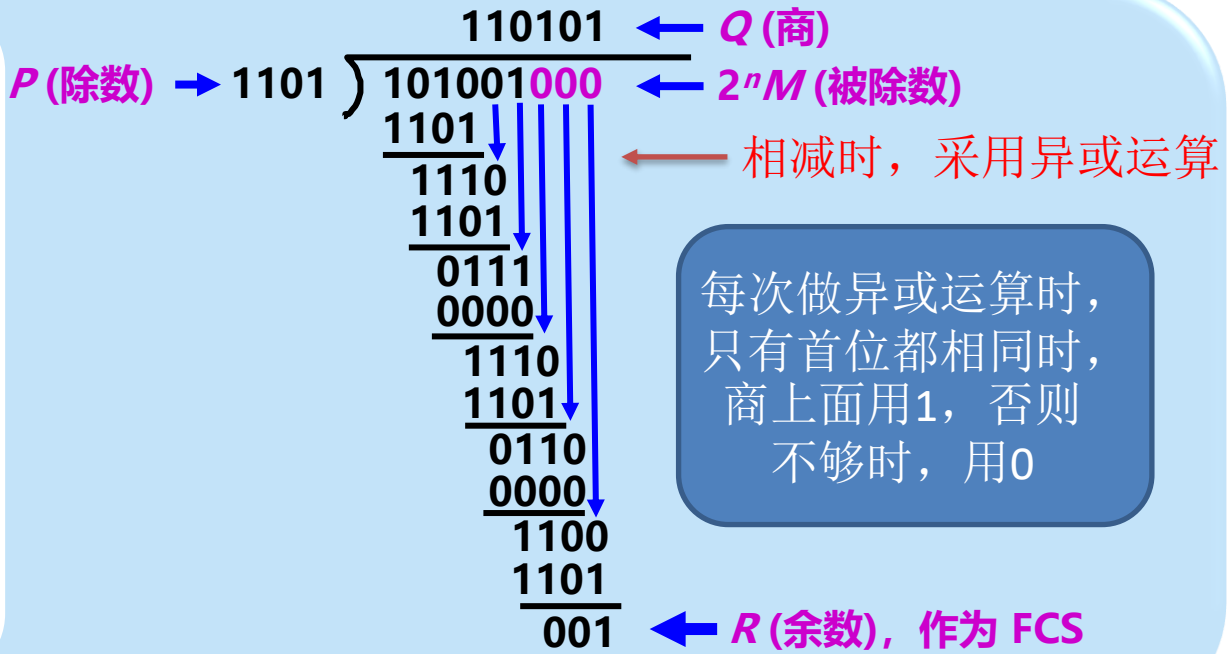
冗余码的计算举例

- 现在 $k = 6$, $M = 101001$。
- 设 $n = 3$, 除数 $P = 1101$,
- 被除数是 $2^nM = 101001000$。
- 模 2 运算的结果是：商 $Q = 110101$，余数 $R = 001$。
- 把余数 R 作为冗余码添加在数据 M 的后面发送出去。发送的数据是：$2^nM + R$，即：101001001，共 $(k + n)$ 位。

循环冗余检验的原理说明

原始数据 M = 101001
除数 P = 1101
做运算时:
被除数: $M * 2^n$
除数: （n+1）位
余数: n位

最后发送的数据=原始数据+余数
发送数据 = 101001001



帧检验序列 FCS

- 在数据后面添加上的冗余码称为**帧检验序列** FCS (Frame Check Sequence)。
- 循环冗余检验 CRC 和帧检验序列 FCS **并不等同**。
  1. CRC 是一种常用的检错方法，而 FCS 是添加在数据后面的冗余码。
  2. FCS 可以用 CRC 这种方法得出，但 CRC 并非用来获得 FCS 的唯一方法。

应当注意

- 仅用循环冗余检验 CRC 差错检测技术只能做到**无差错接受** (accept)。
- **“无差错接受”是指**：“凡是接受的帧（即不包括丢弃的帧），我们都能以非常接近于 1 的概率认为这些帧在传输过程中没有产生差错”。
- 也就是说：“凡是接收端数据链路层接受的帧都没有传输差错”（有差错的帧就丢弃而不接受）。
- **CRC也存在误差，除数越大，检测误差率越小**
- **单纯使用 CRC 差错检测技术不能实现“无差错传输”或“可靠传输”。**

CRC是一种“无比特差错”，而不是“无传输差错”的检测机制
当接收端检测到有错误后，直接丢弃，不要求发送方重发
数据链路层只负责无比特差错的接收

应当注意

- 应当明确，“无比特差错”与“无传输差错”是不同的概念。
- 在数据链路层使用 CRC 检验，能够实现无比特差错的传输，但这还不是可靠传输。
- 要做到“无差错传输”（即发送什么就收到什么）就必须再加上确认和重传机制。（由高层实现）
- 本章介绍的数据链路层协议都不是可靠传输的协议。

A，B通信时，当路由器收到包后，进行CRC检测，若有错，直接丢弃，所以会发生丢包，当B端收到后，B的高层（如：传输层）会要求A重发丢失的数据包，所以在路由器之间并不要求重传，也体现了端到端的通信。

纠错码-海明码

- 码距：
  - 一个编码系统中任意两个合法编码（码字）之间至少有多少个二进制位不同
- 例1：
  - 现有一套编码：00 01 10 11
  - 至少有1个二进制位不同(00 与 01 等)，码距为：1
- 例2：
  - 现有一套编码： 00 11
  - 至少有2个二进制位不同(00 与 11 等)，码距为：2

码距和纠错检错的关系

- **编码的纠错、检错能力与编码的最小距离有关**

$$L - 1 = D + C \ (D \geq C)$$

  - L: 编码的最小距离
  - D: 检测错误的位数
  - C: 纠正错误的位数
- **当L= 3时:**
  - **这种编码最多能检错2位**
  - **这种编码能检错1位、纠错1位**
- **增大L，提高检错和纠错的能力**

码距越大，纠错能力越强，但数据冗余也越大，即编码效率低了。所以，选择码距要取决于特定系统的参数。数字系统的设计者必须考虑信息发生差错的概率和该系统能容许的最小差错率等因素

海明码

- 海明在1950年提出一种编码来纠正单比特错的编码。该编码是将码字内的位从左到右依次编号，编号为2的幂的位是校验位（如第1，2，4，8...），其余为信息位。
- 每个校验位的取值应使得包括自己在内的一些集合服从规定的奇偶性。
- 集合的选取如下：
  - 对编号为K的信息位来说，K可以分解成2的幂的和，如编号为11，11=1+2+8，即第11位由1，2，8校验位校验，它同时属于1，2，8所在的集合。

海明码

- **原则：**
  - 海明码只能纠正1位错
  - 海明码默认进行**偶校验**(除非特殊说明使用奇校验)。
  - 海明码是一串由0和1组成的序列
- **海明码公式：**

$$m + r + 1 \leq 2^r$$

  - m: 原数据的位数
  - r: 校验的位数 **(求出最小的 r)**
- **码字的位排序：**
  - 2的幂次方的位(1,2,4,8,16等)是**校验位**
  - 其他位(3,5,6,7,9)用来填充m个**数据位**

纠正单比特错的校验位下界

- 设计一种编码，它有m个信息位和r个校验位，当r满足什么条件时，能纠正所有单比特错？
- 码字长度n=m+r, 总共有2^n个种组合，其中有效码字数为2^m

每一个有效码字X，

例如X=10101010

对其某一位取反

有n个与其距离为1的无效码字

X1=00101010，X2= 11101010

X3=10001010， X4=10111010

X5=10100010， X6=10101110

X7=10101000， X8=10101011

要纠正单比特错，需采用距离至少为3的编码

（n+1）$2^m<=2^n$，n=m+r代入

$2^r>=n+1$

纠正单比特误码的校验位下界

纠正单比特错的校验位下界

- 设计一种编码，它有m个信息位和r个校验位，当r满足什么条件时，能纠正所有单比特错？
  - 对2^m个有效码字的任何一个而言，有n个与该码字距离为1的无效码字，所以2^m个有效码字每一个都对应有n+1个各不相同的位图，n位码字的总的位图是2^n个。

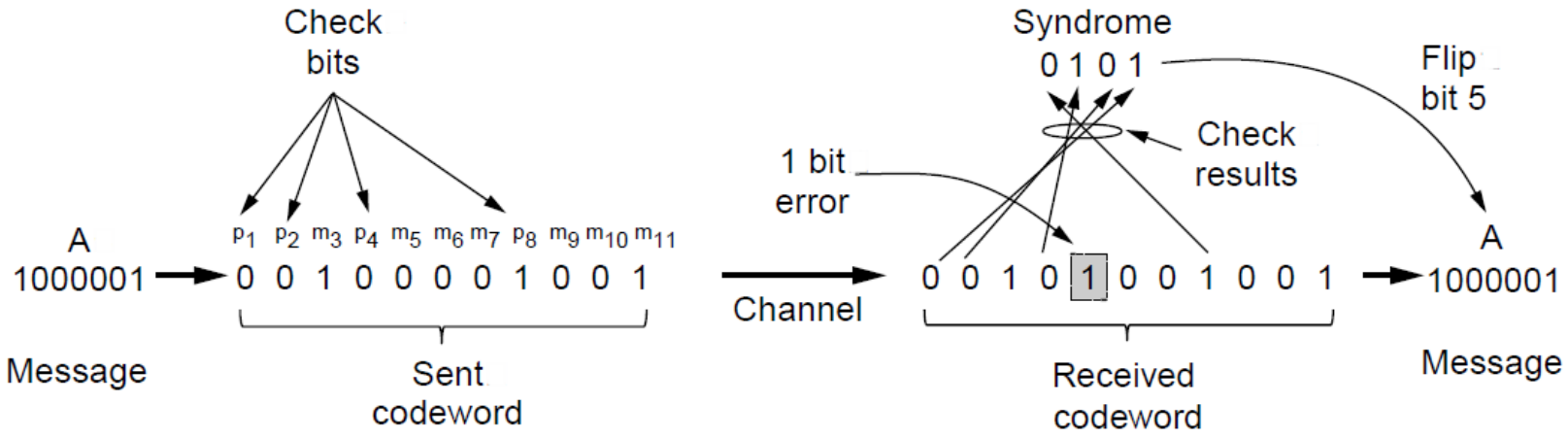
（n+1）$2^m <= 2^n$，n=m+r代入

（m+r+1）$2^m <= 2^{m+r}$

$2^r >= n+1$　纠正单比特误码的校验位下界

$$m + r + 1 \leq 2^r$$

海明码示例



数据信息：1000001(m = 7)

海明码求解

数据信息：1000001(m = 7)

1.算出校验位数r

- **正常情况下我们需要如下此操作：**
- **2^r >= r + m+ 1**
- **这里等于 r =4**

2.确定校验位在海明码中的位置

- **这里按2^r次幂留出来，就像1，2，4，8，16，32。**

H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
		1		0	0	0		0	0	1

3.分组

- 我们需要确认H1,H2,H4,H8这四个校验位都来校验哪些位置。
- 我们按这个规则进行分配，将剩下的数据位分别用4位(r位)二进制表示出来

	H8	H4	H2	H1
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

与H8相关的：H9,H10,H11
与H4相关的：H5,H6,H7
与H2相关的：H3,H6,H7,H10,H11
与H1相关的：H3,H5,H7,H9,H11
因此：
H1 负责 1 3 5 7 9 11 位数的校验
H2 负责 2 3 6 7 10 11 位数的校验
H4 负责 4 5 6 7 位数的校验
H8 负责 9 10 11 位数的校验

4.求出校验位是0还是1

- H1 负责 1 3 5 7 9 11 位数的校验；H2 负责 2 3 6 7 10 11 位数的校验
- H4 负责 4 5 6 7 位数的校验；H8 负责 9 10 11 位数的校验

H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
		1		0	0	0		0	0	1

- 这张表，我们根据偶校验很容易就求出以下结论：
- H3,H5,H7,H9, H11 1的个数为偶数 因此H1=0
- H3,H6,H7,H10,H11 1的个数为偶数 因此H2=0
- H5,H6,H7 1的个数为偶数 因此H4=0
- H9,H10,H11 1的个数为奇数 因此H4=1；至此我们得出了完整的海明码：

H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
0	0	1	0	0	0	0	1	0	0	1

5.判断接收方收到的码字是否出错，以及哪一位出错

- 收到的码字：00101001001
- 按照上述步骤，求出对应的海明码：

H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
1	0	1	1	1	0	0	1	0	0	1

H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11
0	0	1	0	0	0	0	1	0	0	1

- 由上述的两张表，可以看出：

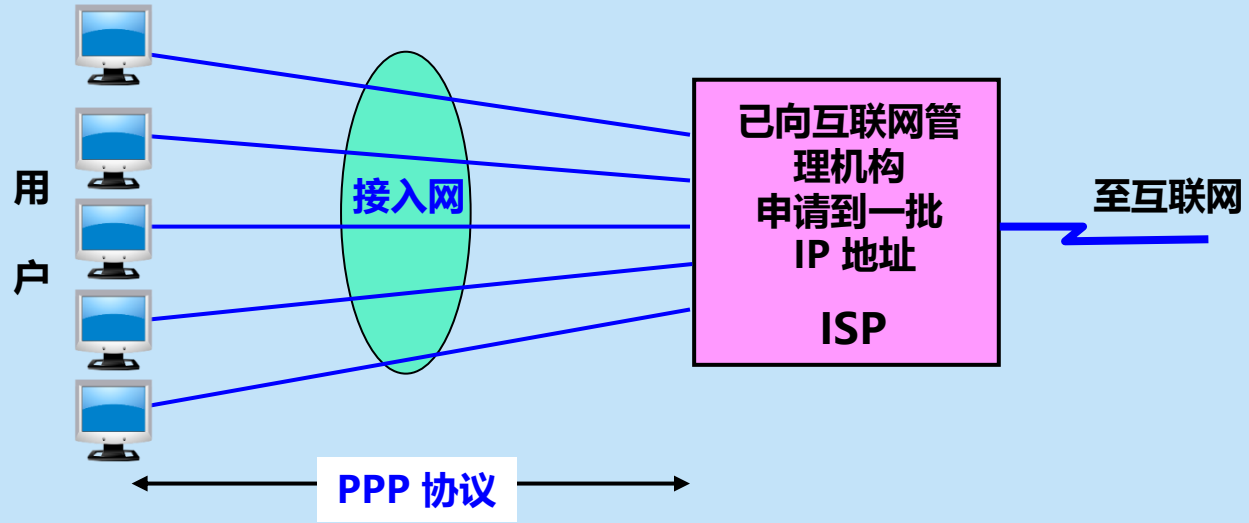
  H1和H4 不同，所以码字中第5（4+1）位出现错误

3.2 点对点协议 PPP

3.2.1 PPP 协议的特点

- 对于点对点的链路，目前使用得最广泛的数据链路层协议是点对点协议 PPP (Point-to-Point Protocol)。
- PPP 协议在 1994 年就已成为互联网的正式标准。

用户到 ISP 的链路使用 PPP 协议



1. PPP 协议应满足的需求

- 简单 —— 这是首要的要求。
- 封装成帧 —— 必须规定特殊的字符作为帧定界符。
- 透明性 —— 必须保证数据传输的透明性。
- 多种网络层协议 —— 能够在同一条物理链路上同时支持多种网络层协议。
  （网络层协议可以是IP, IPX,IPv6 等， PPP支持对这些协议的标识 ）
- 多种类型链路 —— 能够在多种类型的链路上运行。
  （多种类型链路是指：光纤，同轴电缆，双绞线，电话线，可在多种媒体介质上传输）
- 差错检测 —— 能够对接收端收到的帧进行检测，并立即丢弃有差错的帧。

1. PPP 协议应满足的需求（续）

- 检测连接状态 —— 能够及时自动检测出链路是否处于正常工作状态。
  （因不同的情形显示不同的状态，欠费、登录账号密码错误、网线没有插好等，提供各种状态报告）
- 最大传送单元 —— 必须对每一种类型的点对点链路设置最大传送单元 MTU 的标准默认值，促进各种实现之间的互操作性。
- 网络层地址协商 —— 必须提供一种机制使通信的两个网络层实体能够通过协商知道或能够配置彼此的网络层地址。
- 数据压缩协商 —— 必须提供一种方法来协商使用数据压缩算法。
  （在数据传输前，进行压缩，节省带宽）

2. PPP 协议不需要的功能

- 纠错
- 流量控制
- 序号
- 多点线路
- 半双工或单工链路

3. PPP 协议的组成

- PPP 协议有三个组成部分：
  1. 一个将 IP 数据报封装到串行链路的方法。（支持异步/同步串行）
  2. 链路控制协议 LCP (Link Control Protocol)。

     建立并维护数据链路连接，负责身份验证
  3. 网络控制协议 NCP (Network Control Protocol)。

     允许点到点连接上使用多种网络层协议

     （只有当LCP工作正常后，NCP才能通信）

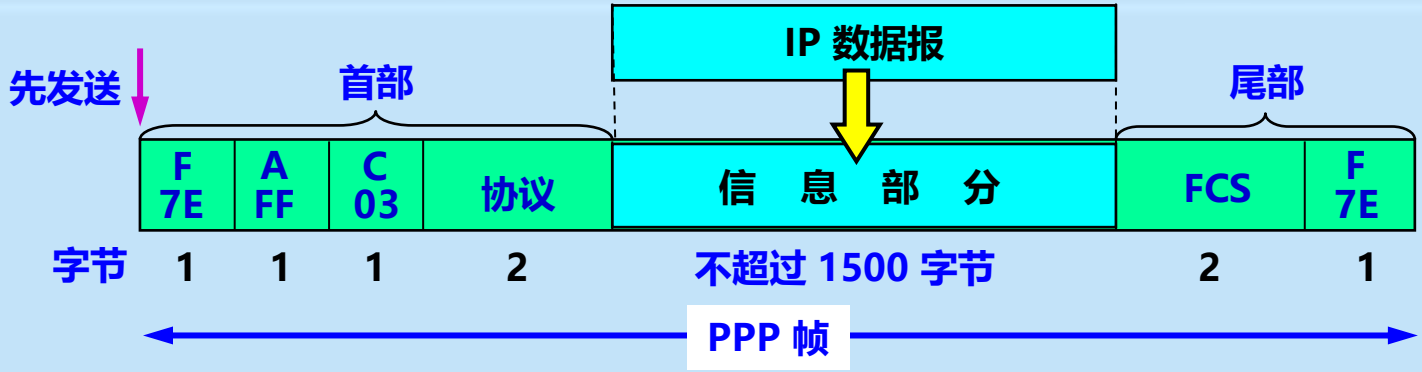
     （例如拨号上网，只有不欠费，账号密码正确之后，才能从ISP获取到IP地址，然后你的网络才能通信）

3.2.2 PPP 协议的帧格式

- PPP 帧的首部和尾部分别为 4 个字段和 2 个字段。
- 标志字段 F = 0x7E （符号“0x”表示后面的字符是用十六进制表示。十六进制的 7E 的二进制表示是 01111110）。
- 地址字段 A 只置为 0xFF。地址字段实际上并不起作用。
- 控制字段 C 通常置为 0x03。
- PPP 是面向字节的，所有的 PPP 帧的长度都是整数字节。

在PPP协议中：第一个字节和最后的字节是相同的，都是0x7E，标志字段A：目标地址，但在PPP协议中不起作用，因为是点到点通信，中间没有其他节点，发送方发送，接收端就能接收，无需特意指定地址

PPP 协议的帧格式



PPP 有一个 2 个字节的协议字段。其值

- 若为 0x0021，则信息字段就是 IP 数据报。
- 若为 0x8021，则信息字段是网络控制数据。
- 若为 0xC021，则信息字段是 PPP 链路控制数据。
- 若为 0xC023，则信息字段是鉴别数据。
- 若为 0xC025，则信息字段是LQR
- 若为 0xC223，则信息字段是安全性认证CHAP

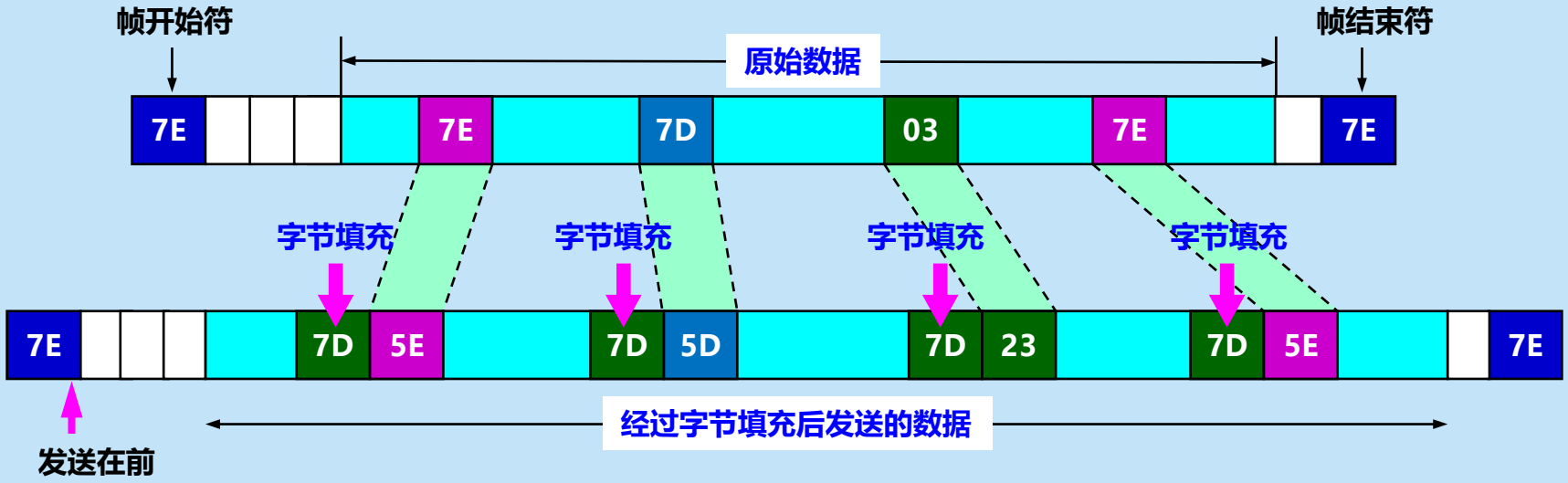
透明传输问题

- 当 PPP 用在异步传输时，就使用一种特殊的字符填充法。
- 当 PPP 用在同步传输链路(SONET/SDH)时，协议规定采用硬件来完成比特填充（和 HDLC 的做法一样）。

字符填充

- 将信息字段中出现的每一个 0x7E 字节转变成为 2 字节序列 (0x7D, 0x5E)。
- 若信息字段中出现一个 0x7D 的字节, 则将其转变成为 2 字节序列 (0x7D, 0x5D)。
- 若信息字段中出现 ASCII 码的控制字符（即数值小于 0x20 的字符），则在该字符前面要加入一个 0x7D 字节，同时将该字符的编码加以改变。

字符填充法


帧开始符
帧结束符
原始数据
7E
7E
7D
03
7E
7E
字节填充
字节填充
字节填充
字节填充
7E
7D
5E
7D
5D
7D
23
7D
5E
7E
经过字节填充后发送的数据
发送在前

零比特填充

- PPP 协议用在 SONET/SDH 链路时，使用同步传输（一连串的比特连续传送）。这时 PPP 协议采用零比特填充方法来实现透明传输。
- 在发送端，只要发现有 5 个连续 1，则立即填入一个 0。
- 接收端对帧中的比特流进行扫描。每当发现 5 个连续1时，就把这 5 个连续 1 后的一个 0 删除。

零比特填充

信息字段中出现了和标志字段 F 完全一样的 8 比特组合 0x7E

0 1 0 0 1 1 1 1 1 1 0 0 0 1 0 1 0

会被误认为是标志字段 F

发送端在 5 个连 1 之后填入比特 0 再发送出去

0 1 0 0 1 1 1 1 1 0 1 0 0 0 1 0 1 0

发送端填入 0 比特

接收端把 5 个连 1 之后的比特 0 删除

0 1 0 0 1 1 1 1 1 0 1 0 0 0 1 0 1 0

接收端删除填入的 0 比特

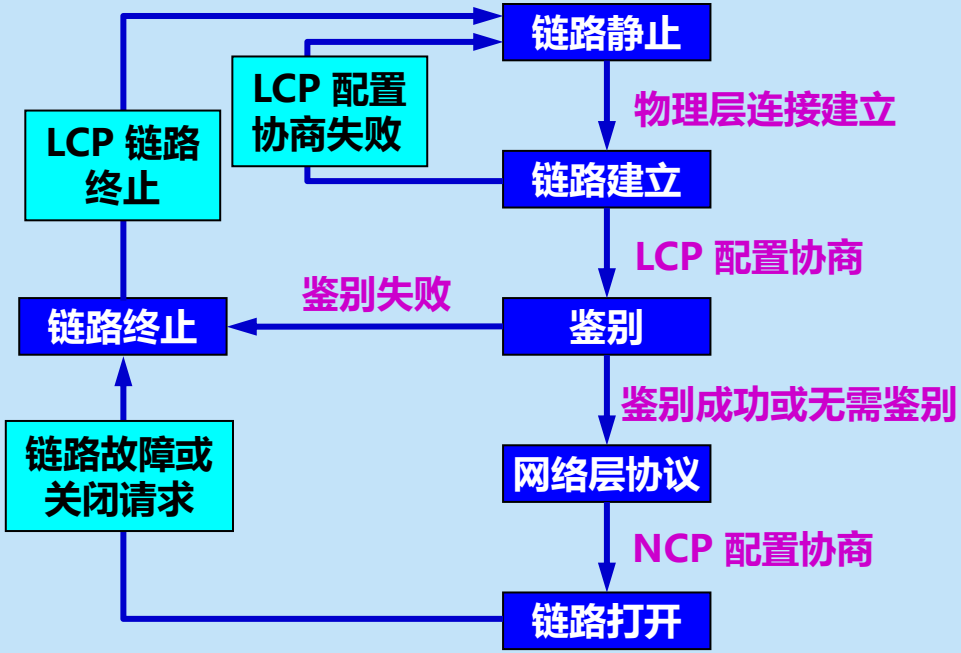
数据部分恰好出现与 0x7E 一样的二进制位串

不提供使用序号和确认的可靠传输

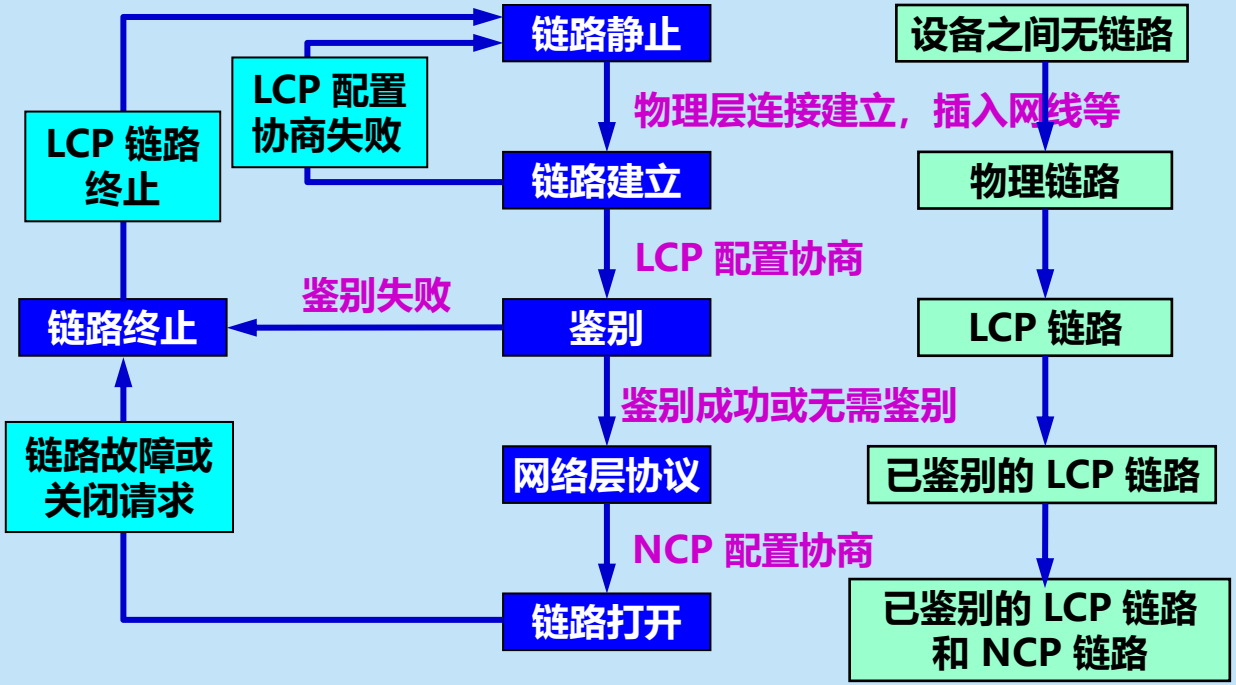
- PPP 协议之所以不使用序号和确认机制是出于以下的考虑：
  1. 在数据链路层出现差错的概率不大时，使用比较简单的 PPP 协议较为合理。
  2. 在因特网环境下，PPP 的信息字段放入的数据是 IP 数据报。数据链路层的可靠传输并不能够保证网络层的传输也是可靠的。
  3. 帧检验序列 FCS 字段可保证无差错接受。

3.2.3 PPP 协议的工作状态

- 当用户拨号接入 ISP 时，路由器的调制解调器对拨号做出确认，并建立一条物理连接。
- PC 机向路由器发送一系列的 LCP 分组（封装成多个 PPP 帧）。
- 这些分组及其响应选择一些 PPP 参数，并进行网络层配置，NCP 给新接入的 PC 机分配一个临时的 IP 地址，使 PC 机成为因特网上的一个主机。
- 通信完毕时，NCP 释放网络层连接，收回原来分配出去的 IP 地址。接着，LCP 释放数据链路层连接。最后释放的是物理层的连接。
- 可见，PPP 协议已不是纯粹的数据链路层的协议，它还包含了物理层和网络层的内容。

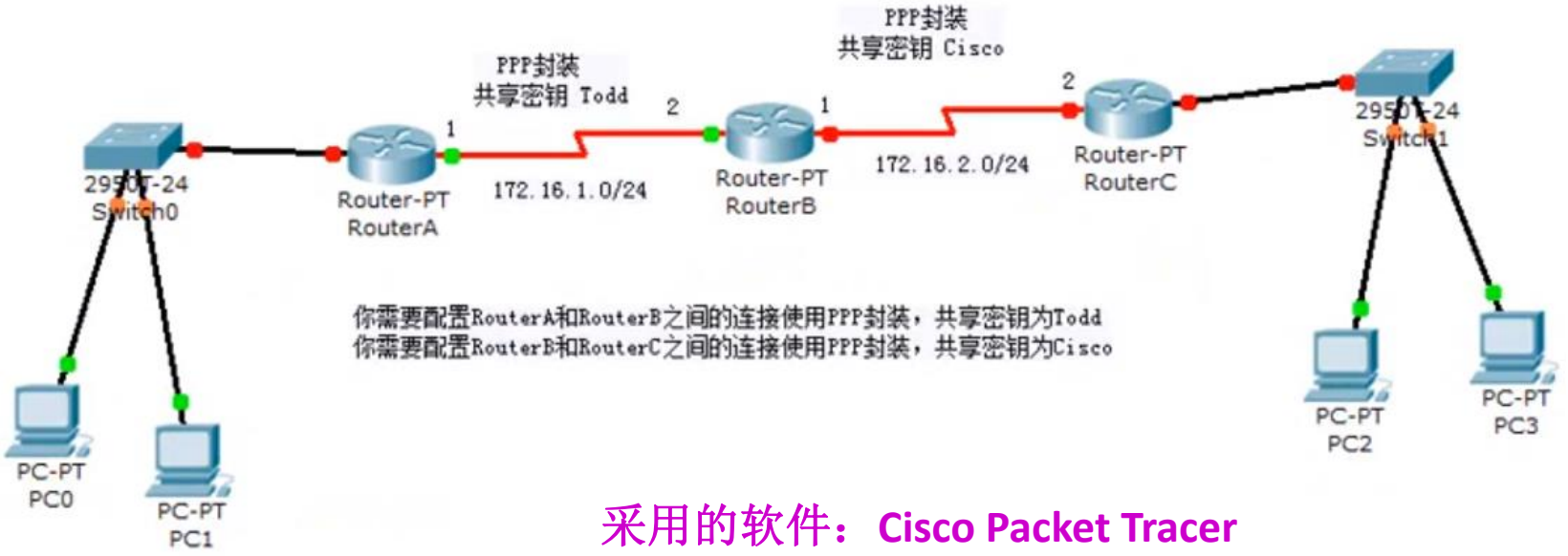

链路静止
物理层连接建立
链路建立
LCP 配置协商
鉴别
鉴别成功或无需鉴别
网络层协议
NCP 配置协商
链路打开
链路故障或关闭请求
链路终止
鉴别失败
LCP 链路终止
LCP 配置协商失败

PPP 协议的状态图


链路静止
设备之间无链路
LCP 配置
协商失败
物理层连接建立，插入网线等
LCP 链路
终止
链路建立
物理链路
LCP 配置协商
鉴别失败
链路终止
鉴别
LCP 链路
鉴别成功或无需鉴别
链路故障或
关闭请求
网络层协议
已鉴别的 LCP 链路
NCP 配置协商
链路打开
已鉴别的 LCP 链路
和 NCP 链路

PPP 协议的状态图

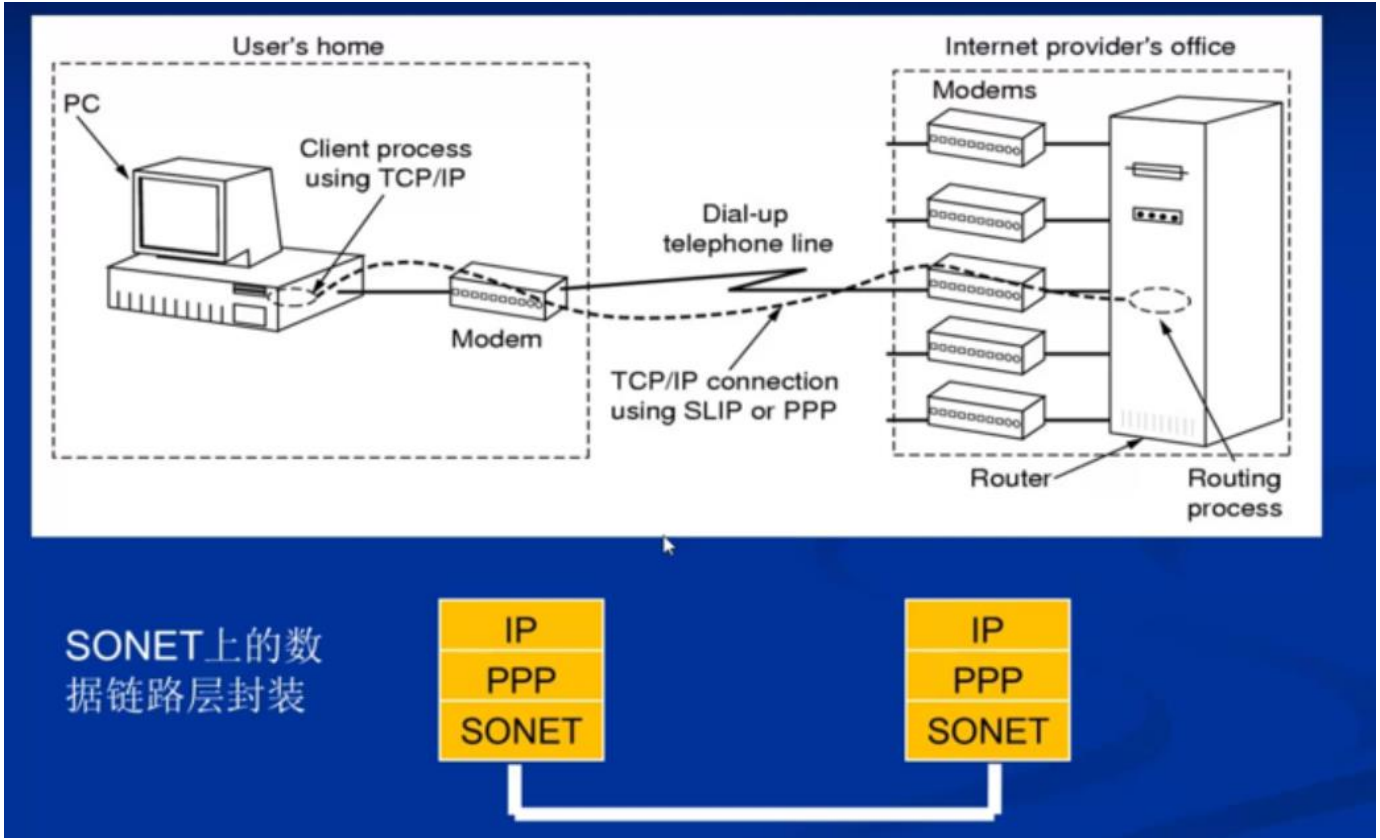
PPP封装演示 ---(视屏P31)



采用的软件：Cisco Packet Tracer

路由器可以封装PPP协议，也可以使用其他协议进行封装
路由器默认采用的封装协议是：HDLC (高级数据链路控制)
rouuterA 和 routerB 相连的接口都需要封装成PPP协议，才能正确通信，还要设置共享密钥

PPP协议


User's home
PC
Client process using TCP/IP
Modem
Dial-up telephone line
TCP/IP connection using SLIP or PPP
Internet provider's office
Modems
Router
Routing process
SONET上的数据链路层封装
IP
PPP
SONET
IP
PPP
SONET

面向位的协议HDLC

- 高级数据链路控制（High-Level Data Link Control）是由国际标准化组织制定的面向位的有序链路层协议。
- 是为非平衡的链路级操作而研制的，采用主从结构，链路上一个主站控制多个从站，主站向从站发命令，从站向主站返回响应。
- HDLC中只有一个地址域，即从站的地址，在命令帧中，它是目的地址，在响应帧中，它是源地址。

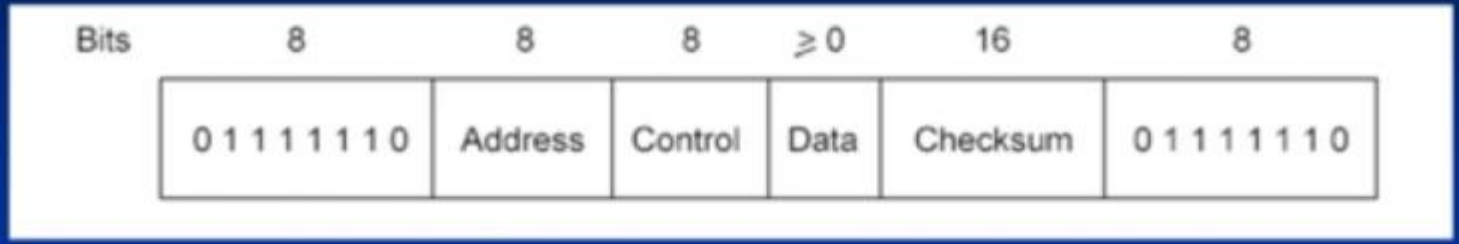
面向位的协议HDLC

(a)不平衡链路结构

(b)平衡链路结构

C：命令； R：相应

HDLC的帧格式



1. 帧标志序列：01111110，作为起始和结束标志，在数据位有5个连续的1出现时，就插入1个0（位填充）
2. 地址段：在命令帧中表示目的地址，在响应帧中表示源地址，全1为广播地址，全0为测试地址

3. 控制字段

用来表示帧的种类，执行信息传送，监控功能：

信息帧（I：Information）：用来实现信息的传送，含有信息字段

监控帧（S：Supervision）：帧中不包含信息字段，具有监控链路的作用，并能对收到的帧进行确认。

无编号帧（U：Unnumbered）：对数据链路进行附加控制。

Bits	1	3	1	3
I帧 (a)	0	Seq	P/F	Next
S帧 (b)	1	0	Type	P/F	Next
U帧 (c)	1	1	Type	P/F	Modifier

Seq：发送端发送序列编号，这里是3比特，采用模8循环编号。

Next：表示发送端准备接收的序列号，也采用模8循环编号。

Type：表示监控功能的类型。Modifier：附加的修改功能。

P/F：在命令帧中作为询问比特，在响应帧中作为终止比特。

监控帧

00——接收就绪(RR),由主站或从站发送。主站可以使用RR型S帧来轮询从站,即希望从站传输编号为N(R)的I帧,若存在这样的帧,便进行传输;从站也可用RR型S帧来作响应,表示从站希望从主站那里接收的下一个I帧的编号是N(R)。

01——拒绝(REJ),由主站或从站发送,用以要求发送方对从编号为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。

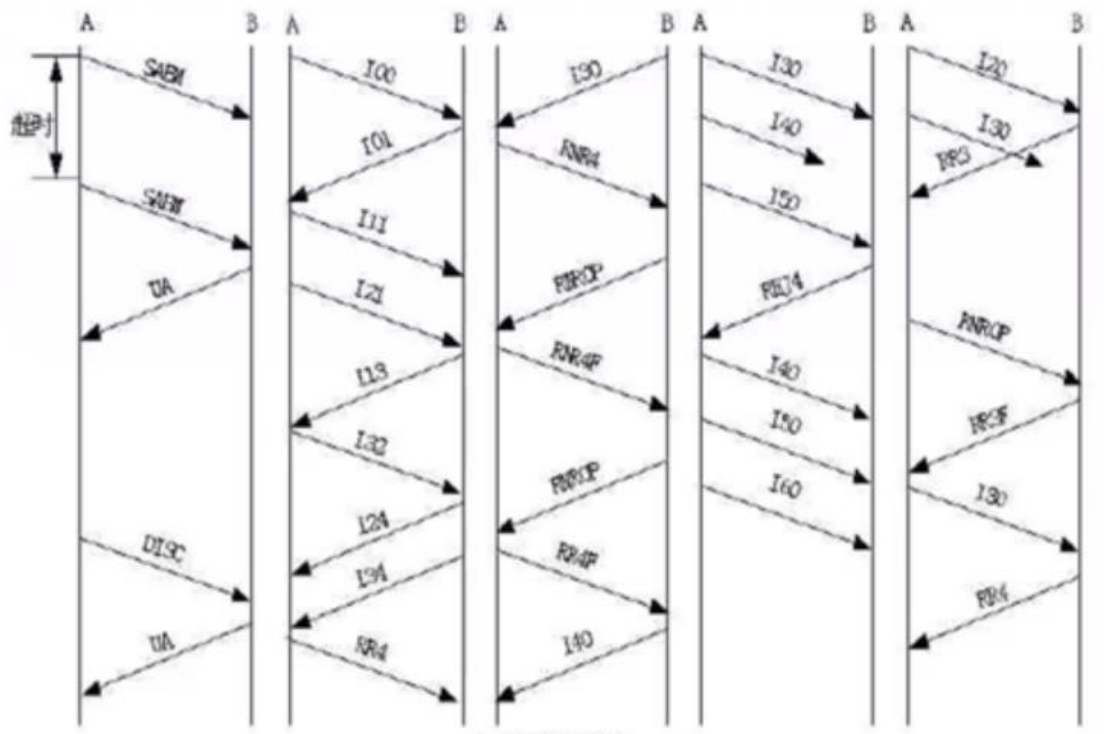
10——接收未就绪(RNR),表示编号小于N(R)的I帧已被收到,但目前正处于忙状态,尚未准备好接收编号为N(R)的I帧,这可用来对链路流量进行控制。

11——选择拒绝(SREJ),它要求发送方发送编号为N(R)单个I帧,并暗示它编号的I帧已全部确认。

无编号帧因其控制字段中不包含编号N(S)和N(R)而得名,简称U帧。U帧用于提供对链路的建立、拆除以及多种控制功能,但是当要求提供不可靠的无连接服务时,它有时也可以承载数据。这些控制功能5个M位(M1、M2、M3、M4、M5,也称修正位)来定义。5个M位可以定义32种附加的命令功能或32种应答功能,但目前许多是空缺的。如:设置数据传输模式SNRM(Set normal response),SABM(Set asynchronous Balanced);请求控制信息Unumbered Poll(UP);终止逻辑链路连接Disconnect(DISC)等。

UA:无编号确认

HDLC的操作示例



HDLC 操作的例

(a) 链路建立和清除 (b) 双向数据交换 (c) 接收站忙 (d) 后退重发 (e) 超时重发

HDLC和PPP的区别

HDLC：

同步链路上封装，每个厂商的HDLC都是私有,不兼容。HDLC用于租用线路的点到点连接,
cisco路由器同步串行链路默认封装协议
cisco的hdlc和标准hdlc帧格式
HDLC缺点:
只支持点到点,不支持点到多点；
不支持IP地址协商；
只能封装在同步链路上，如果是同异步串口的话，只有当同异步串口工作在同步模式下才能使用

PPP：

PPP可以使用在异步串行连接比如拨号或者同步串行连接比如ISDN,PPP优点：
支持同步、异步串行链路
支持多种网络层协议
支持各种连接参数的协商
支持错误检测
允许进行数据压缩

3.3 基本数据链路协议

3.3.1 一种无限制的单工协议

3.3.2 无错信道的单工停-等协议

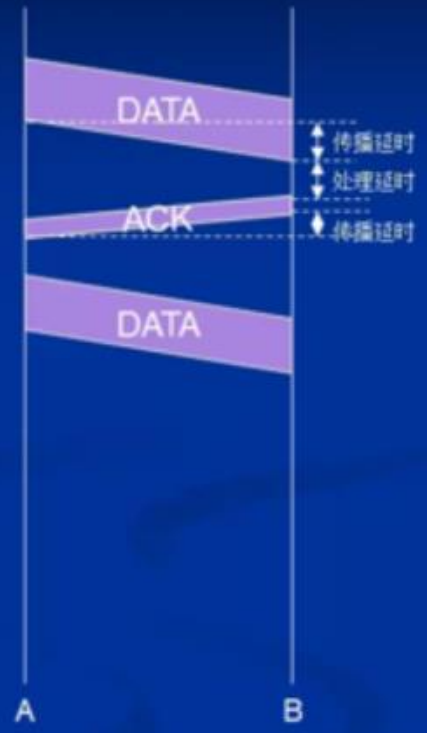
3.3.3 有噪音信道的停-等协议

3.3.1 一种无限制的单工协议

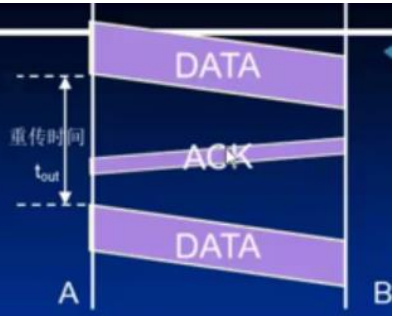
- 完全理想的条件：
  - 数据单向传输，收发双方的网络层一直处于就绪状态，
  - 处理时间可忽略不计，接收缓冲空间无限大（无需任何流量控制）
  - 信道不会损坏或丢失帧（无需任何差错控制）
- 发送端无限循环地重复三个动作：
  - 从网络层取分组。
  - 构造帧。
  - 发出帧。
- 接收端也是无限循环地重复三个动作：
  - 等待事件（唯一的未损坏帧的到达）发生。
  - 帧到达后，从硬件缓冲中取出新到的帧。
  - 将帧的数据部分传给网络层。
- 只需成帧处理，无需做其它任何处理。

3.3.2 无错信道的单工停-等协议

- 条件基本同协议1，
  - 修改：接收端需要一定的接收处理时间，接收缓冲只能存放一个帧。
- 为了防止发送快于接收而造成数据丢失，发送端发送一帧后必须停止发送，等待接收端发回的反馈确认；
- 接收端在收到一个帧并发送网络层后，需向发送端发一反馈确认短帧，表示可发新帧。
- 由于需要反馈，且帧的发送和反馈是严格交替进行的，所以相当于采用半双工信道。



有噪音信道所涉及的问题

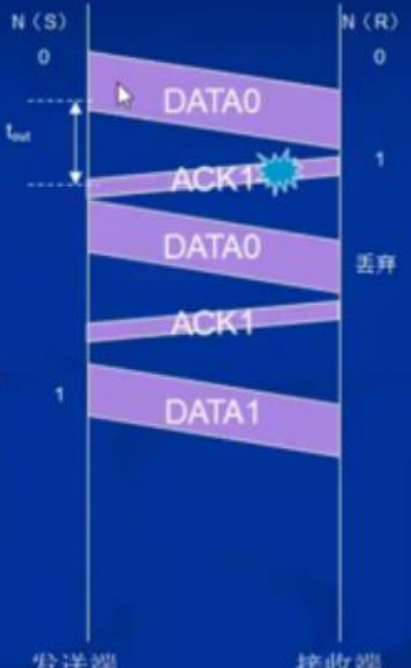


- 考虑实际有噪信道，帧既可能损坏（接收端可通过校验检查出错误），也可能完全丢失。
- 由于帧会丢失，发送端可能收不到反馈的确认帧，因此发送端必须引入超时机制（time out），即增加一个定时计数器，在一定时间后对没有确认的帧进行重发，也称作ARQ（Automatic Retransmit reQuest）。
  - 时间值应选择稍大于两倍端到端的信号传播时间和接收端的接收处理时间之和。
- 由于帧会丢失（假如确认帧丢失），为避免接收端收取重复的数据帧，必须通过为帧编制序号来解决重复帧的问题。
  - 帧的序号位数应尽量的短从而少占用帧头的空间，在简单停-等协议中只需1个比特位（“0”→“1”，“1”→“0”）即可。

3.3.3 有噪音信道的停-等协议

ACKn表示“第n-1号帧已经收到，现在希望接收第n号帧”

- 发送端帧序号$N(S)$表示当前所发帧的序号，接收端维护的帧序号$N(R)$表示接收端当前所期待接收的帧序号。
- 发送端从网络层取得第一个分组组帧，将$N(S)$=0的序号放入帧头中作为第一个帧发送，并启动定时计数器，等待响应帧。
- 接收端收到一个帧后，对其序号和$N(R)$进行比较：
  - 若相等则对其接收，校验正确后送交网络层，将$N(R)$加1（模2运算）并放入确认帧中反馈回发送端；若校验出错，则丢弃错帧，保持$N(R)$不变并放入确认帧中反馈回发送端。
  - 若不等，则将其作为重复帧丢弃；N（R）值不变，反馈确认帧。
- 发送端若在规定的时间内没有收到接收端的反馈确认帧（超时），就认为数据帧丢失，在保持$N(S)$不变的情况下重新发送缓冲器中的（旧）帧；
- 发送端若接收到确认帧后，比较确认帧中的序号和$N(S)$：
  - 若不等，则将确认帧中的序号赋予$N(S)$，从网络层获取新的分组并组成新帧（$N(S)$作为序号放入帧头中）交由物理层发送出去。
  - 若相等，则保持$N(S)$不变，重新发送缓冲器中的（旧）帧；

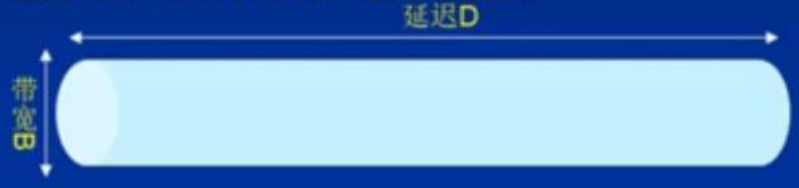


停-等协议对信道利用率的影响

- 在时延大的信道（如卫星通信）中，停-等协议的效率是很低的。
- 考虑卫星通信，典型的传播时间约为270ms。假设一个帧的发送时间为20ms，则从发送站开始发送算起，经20ms+ 270ms=290ms，数据帧才能到达目的站。假设不考虑目的站的处理时间，且认为确认帧非常短，其发送时间可忽略不计，则又需270ms确认帧才能被发送站收到。因此信道的利用率为：20ms/(290ms+270ms)=1/28，非常低。
- 为了提高传输效率，可以设想让发送站连续不断地发送数据帧，当发完第28个帧数据后（290+270），恰好第1帧的确认帧到达，根据确认可紧接着发第29帧或重发第1帧。以后，每过20ms（发一个帧）就有一个确认帧到达，这样信道的利用率就大大地提高了。
- 允许发送站连续发送多个帧而不需等待确认的做法称作管道化（pipelining），属于一种窗口（windows）机制。

延迟 * 带宽乘积

- 单向延迟×带宽——带宽延迟乘积：BD，（单位：比特\字节\帧）
  - 第一个比特到达接收端之前，发送者能发送的比特数，是一个有用的链路性能参数，表示该管道的容量。



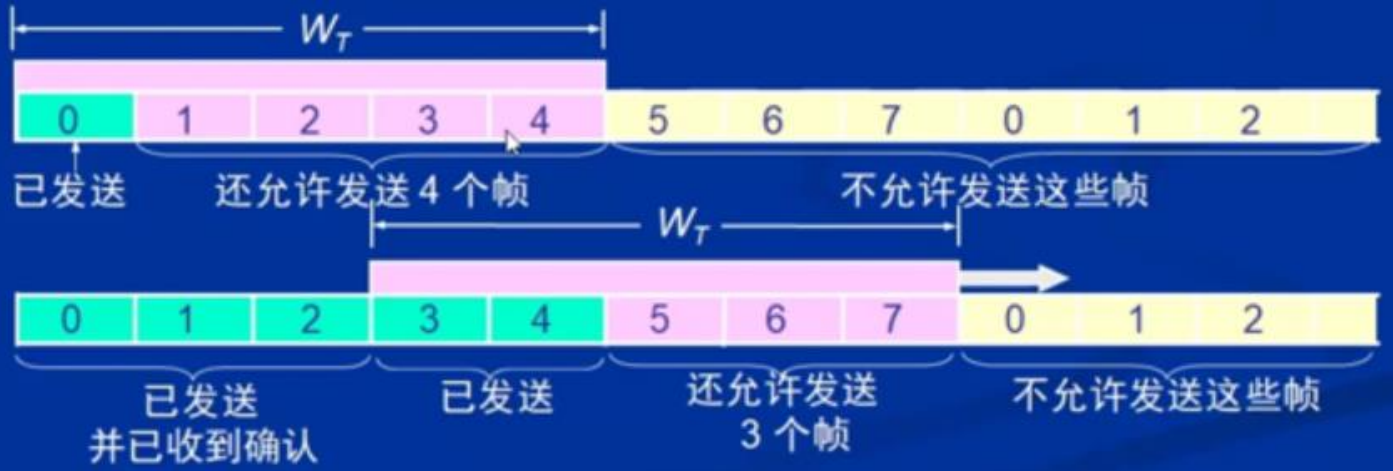
- 往返延迟×带宽： 2BD
  - 假设发送窗口为W，它是发送端在接收到接收端给出的确认之前能够发送的帧（或字节，比特）数。
  - 若要链路利用率100%，则W=2BD+1（单位：帧，注意，这里的BD单位要换成帧），
  - 链路利用率≤W/(1+2BD)，注意，这里的+1是因为必须接收完整帧后确认帧才会被发出，所以W和BD的单位也应该是帧。该值是上限，因为没有考虑处理时间。

3.4 滑动窗口协议

- 滑动窗口协议是一种非常可靠、适用于各种条件的通用流量控制协议，特别是在效率、复杂性及对缓冲区的需求等方面可作灵活调配。
- 发送端和接收端分别设定发送窗口和接收窗口 。
- 发送窗口用来对发送端进行流量控制。
- 主要的滑动窗口协议有回退N协议（go-back-n，出错全部重发）和选择重发协议，停-等协议是窗口大小为1的滑动窗。
- 帧序号用n位编码，范围为$0 \sim 2^n-1$

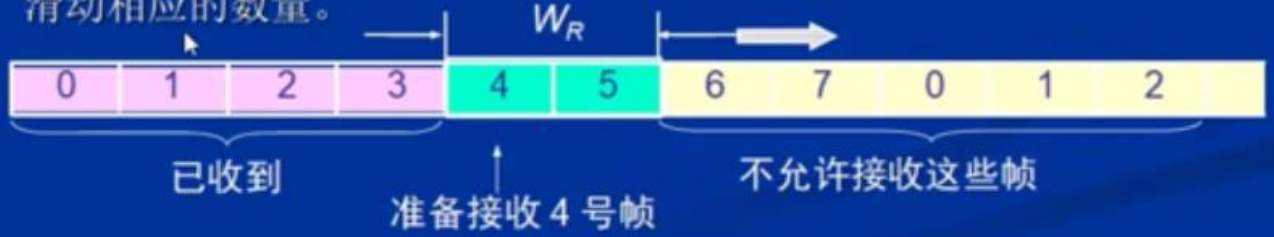
发送窗口

- 发送窗口就是发送端允许不等确认而连续发送的帧的序号表。
- 允许连续发送的帧的数量称为发送窗口尺寸，表示为W_T。发送端必须有W_T个输出缓冲区来存放W_T个数据帧的副本以备数据帧的重发。
- 当发送端收到发送窗口下沿帧的肯定确认时，将发送窗口整体向前滑动一个序号，并从输出缓冲区中将相应的数据帧副本删除。



接收窗口

- 接收窗口是接收端允许接收的帧的序号表。
- 允许接收的帧的数量称为接收窗口尺寸W_R。同样接收端也必须设置相应数量的缓冲区来支持接收窗口。
- 对接收端收到的帧的序号落在接收窗口外的帧被直接丢弃。只有落在接收窗口内的帧才会被接收端进行校验处理，若校验正确：
  - 当接收的帧不是接收窗口下沿帧时，必须暂存在输入缓冲区。
  - 当接收到接收窗口下沿帧时，会将其连同后面连续的若干个检验过的正确帧按顺序交给网络层，在发回确认帧的同时将接收窗口向前滑动相应的数量。

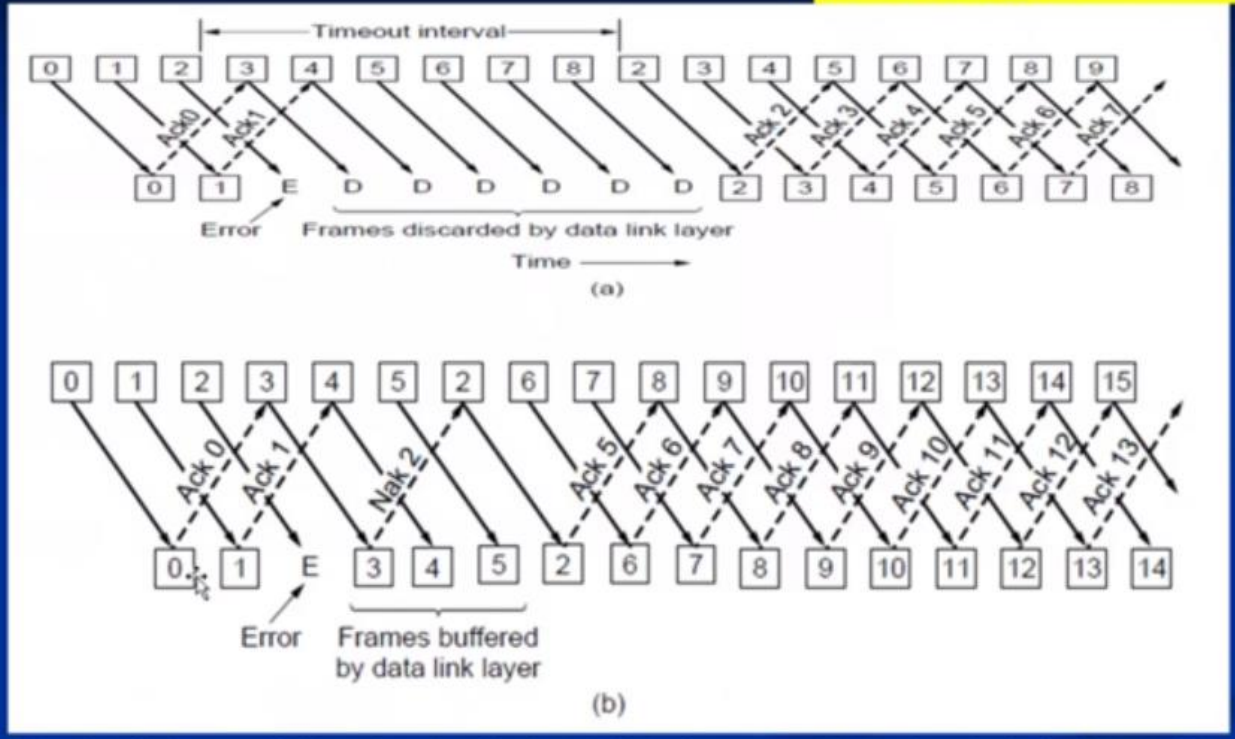


数据的捎带确认

- 在实际通信中，通常收发双方都相互发送数据。
- 为了提高效率，可以将确认信息放在数据帧中作为一个控制字段连同数据一起发送给对方，这种方式称为捎带应答（piggybacking）。
- 当一方收到对方的数据帧后：
  - 若正好也有数据需发给对方，则立即可使用捎带应答。
  - 若暂时没有数据需发给对方或数据还未准备好，则等待一定的时间，如果在该时间内准备好了数据，则可以使用捎带应答。如果未准备好，为了防止对方等待时间过长而超时重发，必须立即发送一个单独的确认帧。
- 使用捎带应答就不用对每一个帧都作确认，可以用对下一帧的期待来代替对该帧之前的所有帧的确认。

主要的滑动窗口协议：回退N协议和选择重发协议

实际网络中大多用ACK*n* 表示“第 *n* – 1 号帧已经收到，现在期望接收第 *n* 号帧”



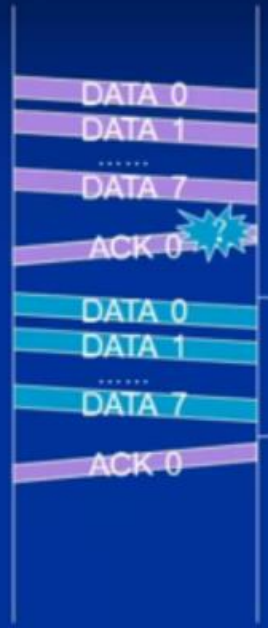
管道化及差错恢复（a）接收方窗口为1时1个错误的影响；（b）接收方窗口很大时1个错误的影响

回退N协议（go-back-n）

- 回退N协议(也叫出错全部重发协议)中，发送窗口的尺寸是大于1，而接收窗口的尺寸则等于1。
- 由于接收窗口的尺寸为1，接收端只能按顺序接受数据帧，一旦某个帧出错或丢失，只能丢弃该帧及其所有的后续帧（因为发送窗口的尺寸是大于1的）。发送端超时后需重发出错或丢失的帧及其后续所有的帧。
- 发送端需要为每个待确认的帧都各自设置一个定时计数器。
- 发送窗口的尺寸不能超过2^n-1(这里的n为序号的编码位数)，否则会造成接收端无法分辨新、旧数据帧。
- 出错全部重发协议只要求发送端保持一定数量的缓存来保存没有确认的数据帧，对接收端没有缓存的要求。但在误码率高的情况下，会大大降低信道的利用率。

发送窗口尺寸超过2^n-1造成帧的混淆

- 设序号位数为3，序号0~7，发送窗口假设=8.
- 发送方考虑两种情况：
  - 所有确认帧都收到，发下一批数据。
  - 所有确认帧都丢失，重传当前窗口所有数据帧。



接收方：

是新数据还是旧数据？

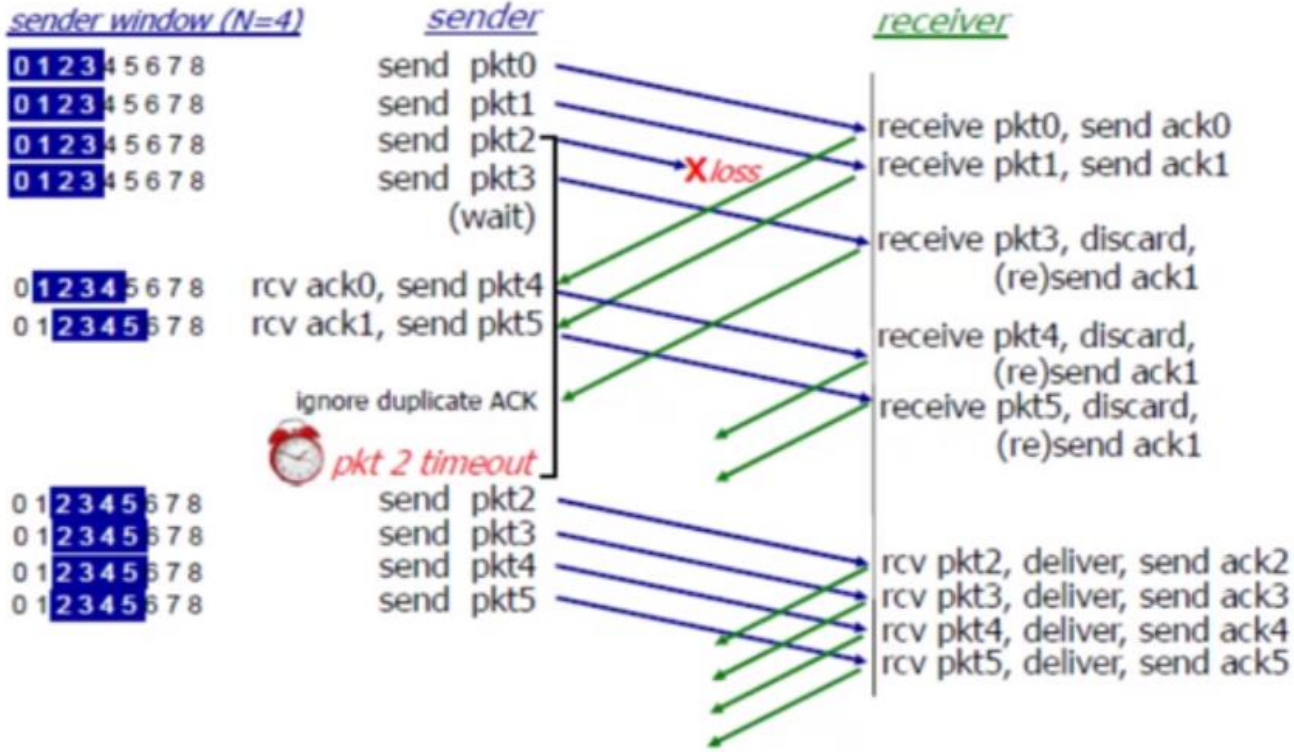
如果前一批确认顺利到达，则是新数据，收；

否则，发送端超时重传，是旧数据，不收。

- 设发送窗口假设=7.

累计确认：当确认序号为n时，意味着n之前的帧如n-1、n-2等帧都自动得到确认。

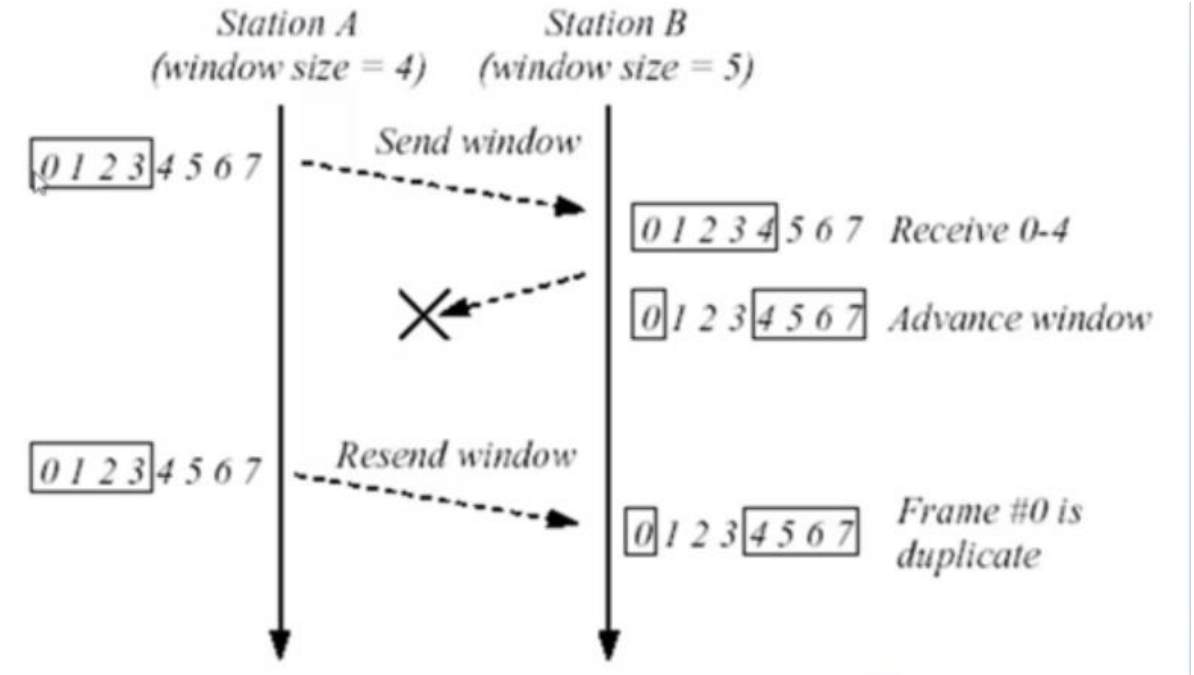
GBN in action



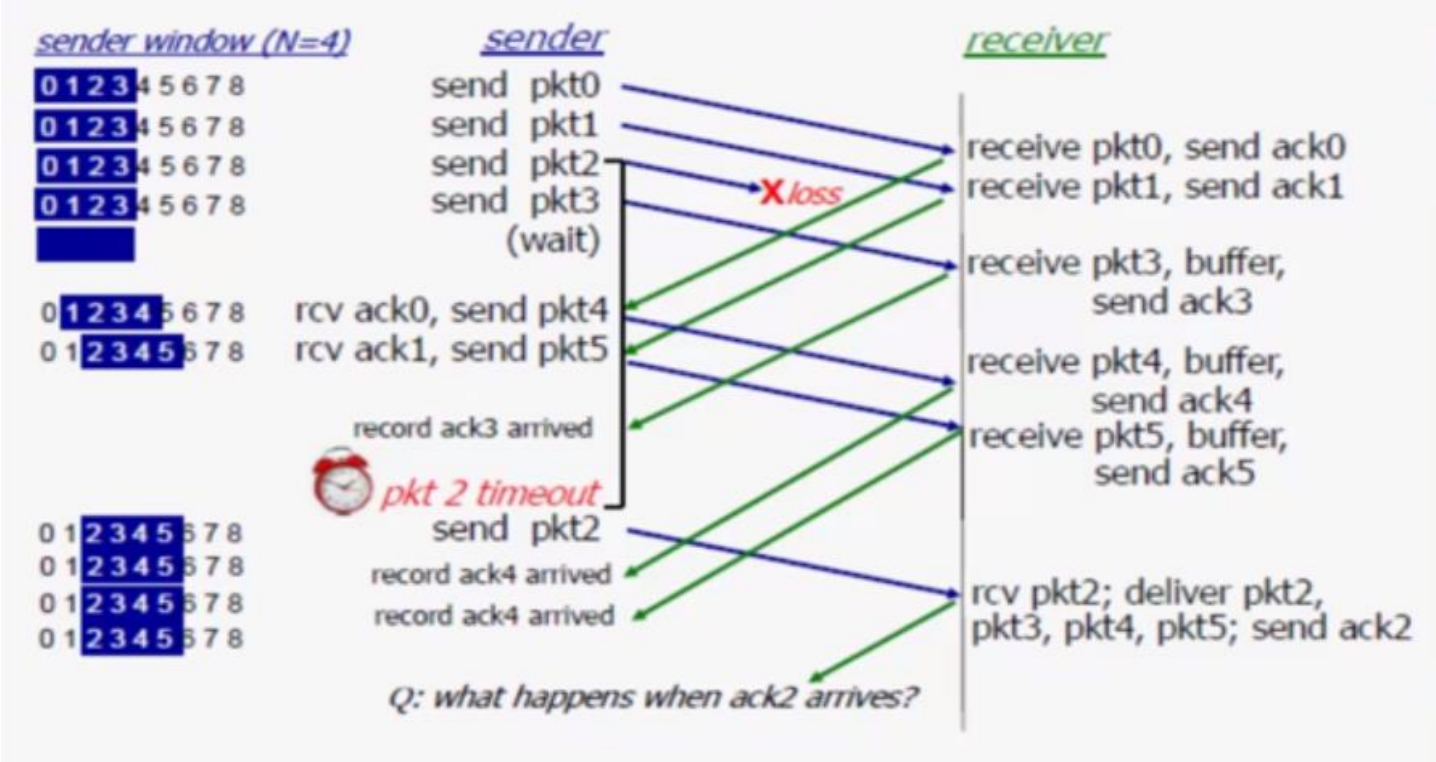
选择重发协议

- 选择重发协议中，发送和接收窗口的尺寸都大于1。
- 由于接收窗口的尺寸大于1，接收端可存储坏帧之后的其它数据帧（落在接收窗口），接收端对错帧发否定确认帧，因此发送端只需重发出错的帧，而不需重发其后的所有后续帧。
- 接收端正确收到重发的帧后，可对其后连续的已接收的正确帧作一次总体确认（最大序号的确认），并交送网络层。大大提高了信道的利用率。
- 接收窗口的尺寸不能超过2^{n-1}（即序号范围的1/2），否则可能造成帧的重叠。
- 发送窗口的尺寸一般和接收窗口的尺寸相同，发送端为每一个输出缓存区设置一个定时计数器，定时器一旦超时，相应输出缓存区中的帧就被重发。

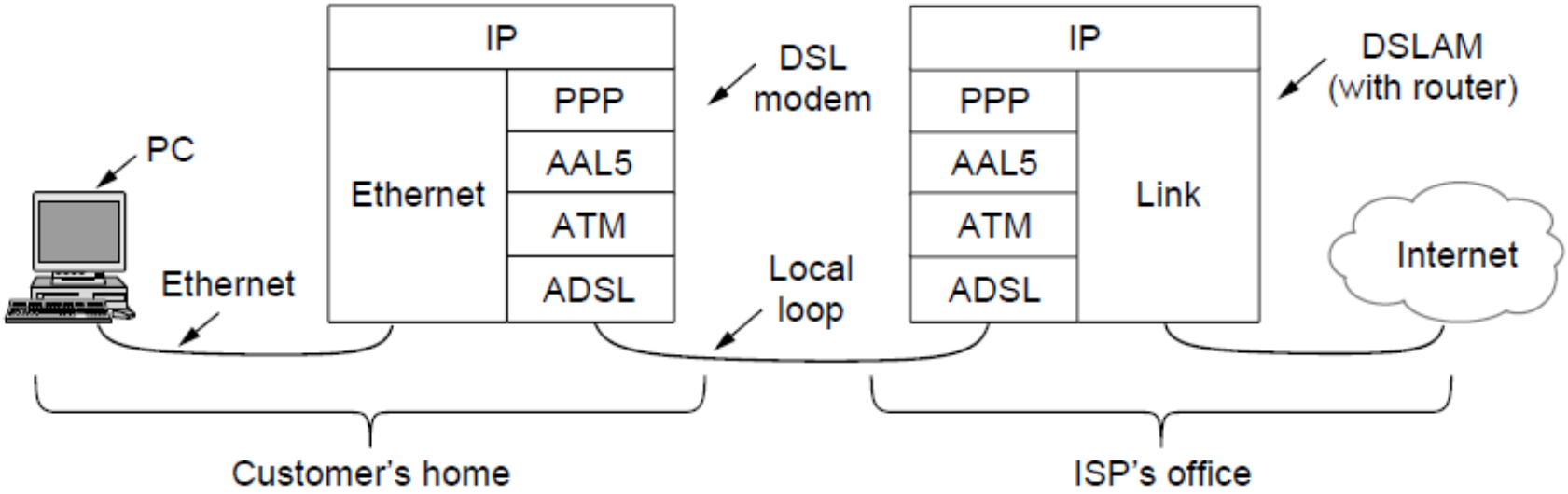
接收窗口的尺寸超过2^{n-1}造成帧的重叠



Selective repeat in action



数据链路协议实例—对称数字用户线



ADSL协议栈

ATM

- ATM即异步转移模式:
- 综合了传输复用交叉连接和交换的技术
- ATM是快速分组交换技术采用统计复用方式
- 一种结合了电路交换和分组交换的各自优点的技术
- ATM能传送所有类型的业务量(话音数据图像等)

传统交换方式的优缺点

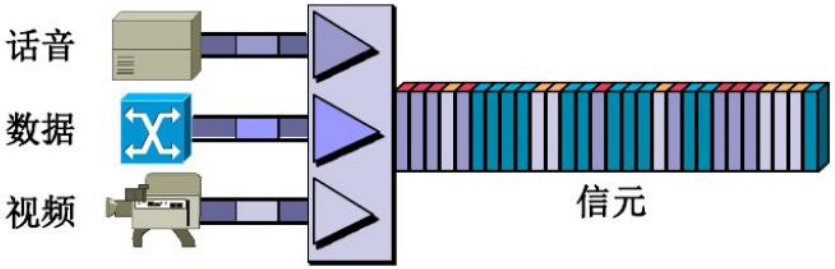
传统交换方式

- 电路交换(Circuit Switching)
  因同一连接的信息经过相同的路由所以时延抖动小
  易于高速率的交换
  采用固定时隙分配电路交换只能支持单一速率
  带宽浪费
- 分组交换(Packet Switching)
  分组交换可以支持多种速率的交换
  系统延迟不确定性
  分组交换的交换速率不易提高

ATM交换方式的特点

- 以信元为单位的异步转移模式异步意味着来自任意用户的信息信元流不必是周期性的便于实现线路带宽的共享和统计复用
- ATM网络能同时传送多种类型的信息承载多种通信业务并且能够提供QoS Quality of Service 服务
- 信元长度固定容易实现硬件交换实现高速交换

ATM的基本特点



- 使用固定长度的信元
- 面向连接
- 采用统计复用方式
- 支持多速率、多业务
- 可适应于局域网和广域网

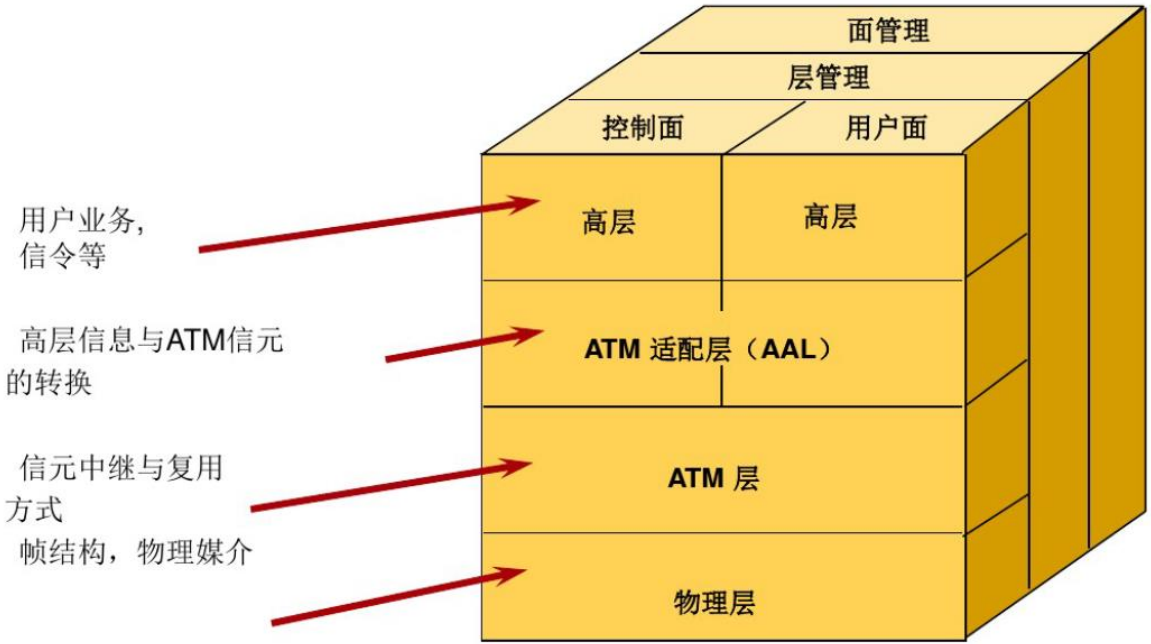
ATM的关键内容

- 采用 53-byte 固定长度的信元 = 48B 净荷 + 5B 信元头

- 信元使用连接标识VPI/VCI通过ATM交换机交换
- 面向连接：在通信前先建立端到端的虚连接 (VCs)

- 建立连接时就定义业务（或连接）的QoS属性

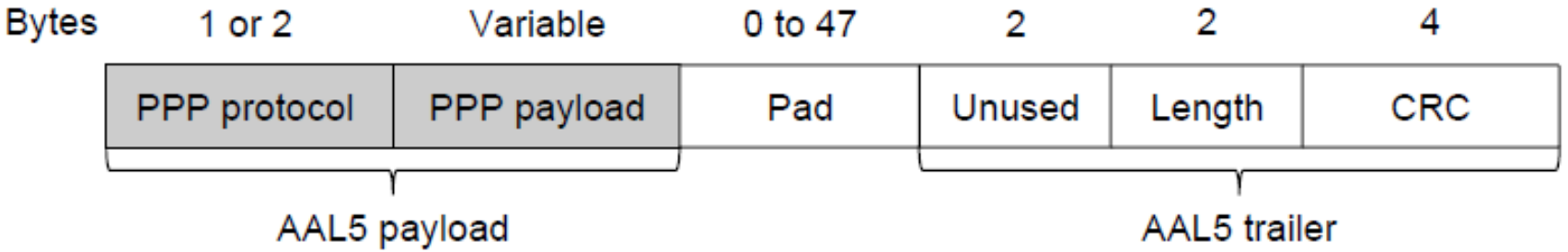
ATM的协议参考模型


面管理
层管理
控制面
用户面
高层
高层
ATM 适配层（AAL）
ATM 层
物理层
用户业务，
信令等
高层信息与ATM信元
的转换
信元中继与复用
方式
帧结构，物理媒介

ATM各层功能

高层		高层功能	
ATM层		一般流量控制 信元头产生与提取 信元VPI/VCI翻译 信元复用和解复用	层管理
AAL	CS	会聚	
	SAR	分段与组装	
物理层	TC	信元速率解耦 HEC序列的产生/检验 信元定界 传输帧适配 传输帧产生/恢复	
	PM	比特定时 物理媒介	

运载PPP数据的AAL5帧



在AAL5帧内，PPP的一下功能不是必需的：

- 成帧功能
- CRC检错功能

小结

总结：

- 面向网络层提供的服务：
  - 面向连接，无连接有确认，无连接无确认
- 成帧
- 差错检测：检测码，纠错码
- 超时重传
- 流量控制：滑动窗口
- 协议：**HDLC，PPP**

作业：

第三章：**3,6,9,11,16,17,20,23**