〇、链路层问题

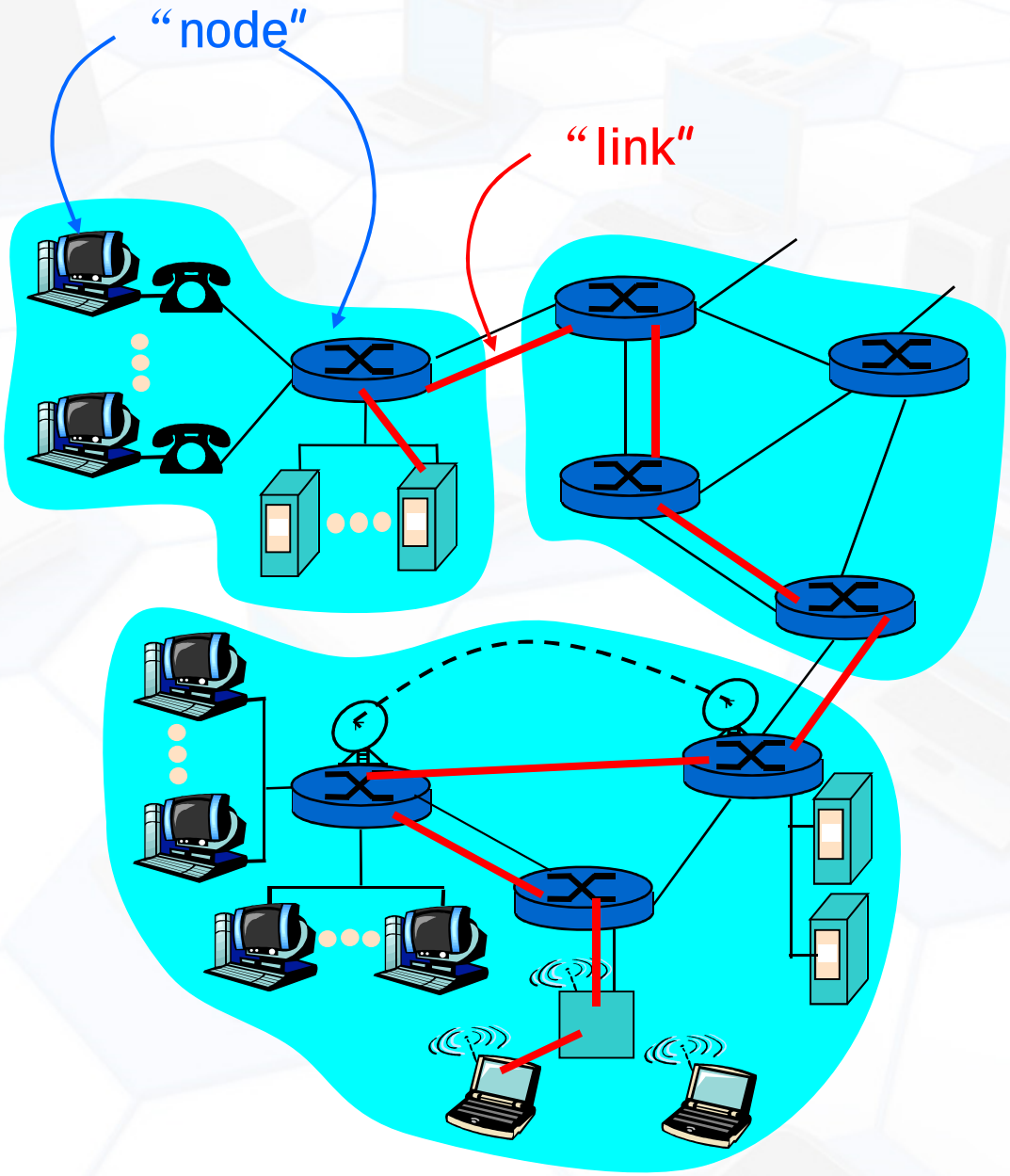
1. 如何理解“在有差错的线路上进行无差错的传输”？

2. 链路层中涉及的Multipe Access是什么概念？是否就是“MAC层”一词中的M和A？

3. 有了链路层协议，物理层是否就可以不需要差错控制了？为什么？

一、链路层功能

- 要解决的问题
  - **如何在有差错的线路上，进行无差错传输**
- ISO关于数据链路层的定义
  - **数据链路层的目的是为了提供功能上和规程上的方法，以便建立、维护和释放网络实体间的数据链路**
- 节点（node）：网络中的主机（host）和路由器（router）称为节点
- 链路（link）：通信路径上连接相邻节点的通信信道称为链路
- 帧（frame）：数据链路层的数据封装，用于包装第三层的数据报（datagram，或分组packet）



数据链路层协议定义了在一条链路的两个节点间交换的数据单元格式，以及节点发送和接收数据单元的动作

§ 数据链路控制规程

Ø 为使数据能迅速、正确、有效地从发送点到达接收点所采用的控制方式

§ 数据链路层协议应提供的最基本功能

Ø 数据在链路上的正常传输（建立、维护和释放）

Ø 数据的定界与同步（成帧，Framing）

Ø 差错控制（检错与纠错）

Ø 流量控制（Flow Control）

§ 为网络层提供三种合理的服务

Ø 无确认无连接服务，适用于

ü误码率很低的线路，错误恢复留给高层

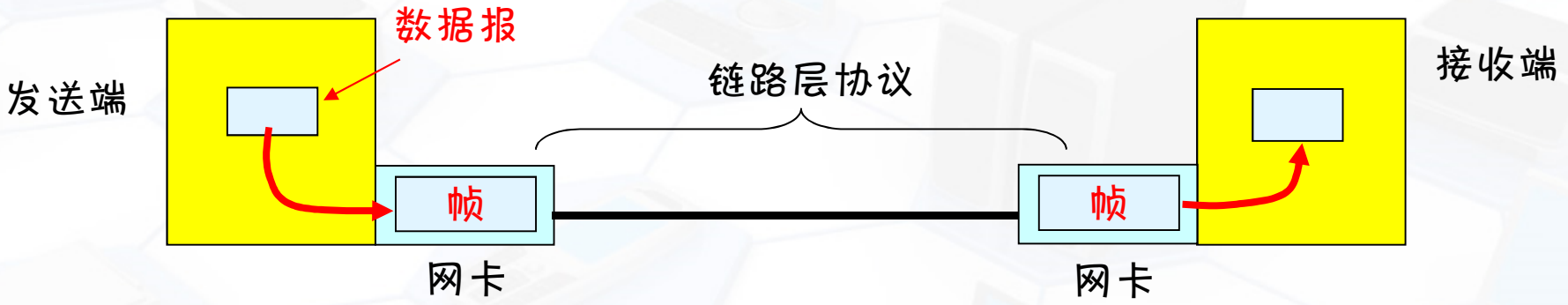
ü实时业务

ü大部分局域网

Ø 有确认无连接服务，适用于不可靠的信道，如无线网

Ø 有确认有连接服务

网卡间的通讯



- 链路层协议在网卡中完成
- 发送方完成：
  - 把数据报包装成数据帧
  - 加入检错位、流量控制信息等

- 接收方工作：
  - 检查是否有错，流量控制等
  - 从数据帧中拆出数据报交给接收方
- 网卡有一定的自主权
- 网卡涵盖了链路层与物理层

二、成帧（Framing）

将比特流分成离散的帧，标识每个帧的起始与结束，并计算每个帧的校验和

成帧方法：

1）字符计数法

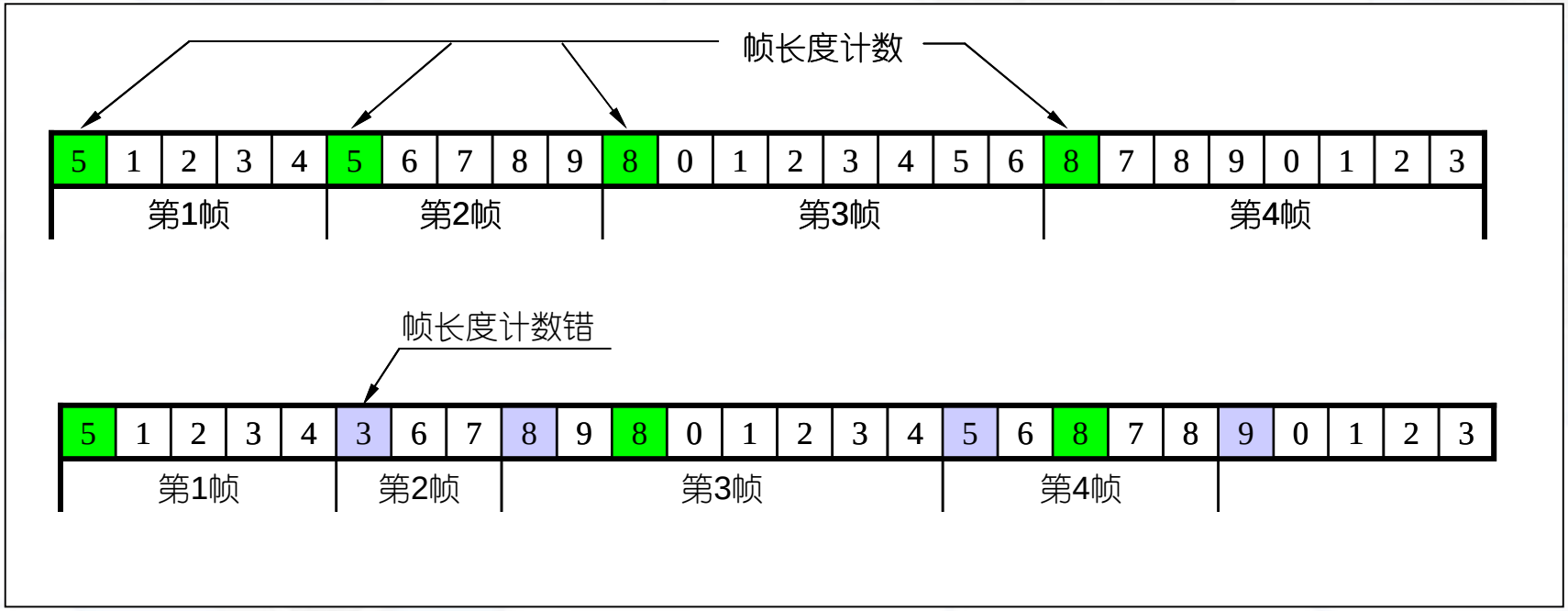
2）带字符填充的首尾字符定界法

3）带位填充的首尾标记定界法

4）物理层编码违例法

1. 字符计数法

把帧的长度用一个字节表示，作为帧的头部



一旦帧长度计数被误读，将不能继续同步，所以很少采用

2. 带字符填充的首尾字符定界法

- ü 起始字符 DLE STX，结束字符DLE ETX字符填充
- ü 缺点：局限于8位字符和ASCII字符传送

§ 用特殊的字符作为帧头和帧尾

如：DLE STX My name is Jone DLE ETX

DLE	STX	M	y		n	a	m	e		i	s		J	o	n	e	DLE	ETX
10	02	4D	79	20	6E	61	6D	65	20	69	73	20	4A	6F	6E	65	10	03

接收方一旦丢失了帧信息，只要查找DLE STX就可重新确定帧边界

如果数据中出现首尾字符，则插入重复的定界符

3. 带位填充的首尾标记定界法

- 帧的起始和结束都用一个特殊的位串“01111110”，称为标记(flag)
- 采用“0”比特插入删除技术

(a) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

(b) 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0

插入的比特

(c) 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0

4. 物理层编码违例法

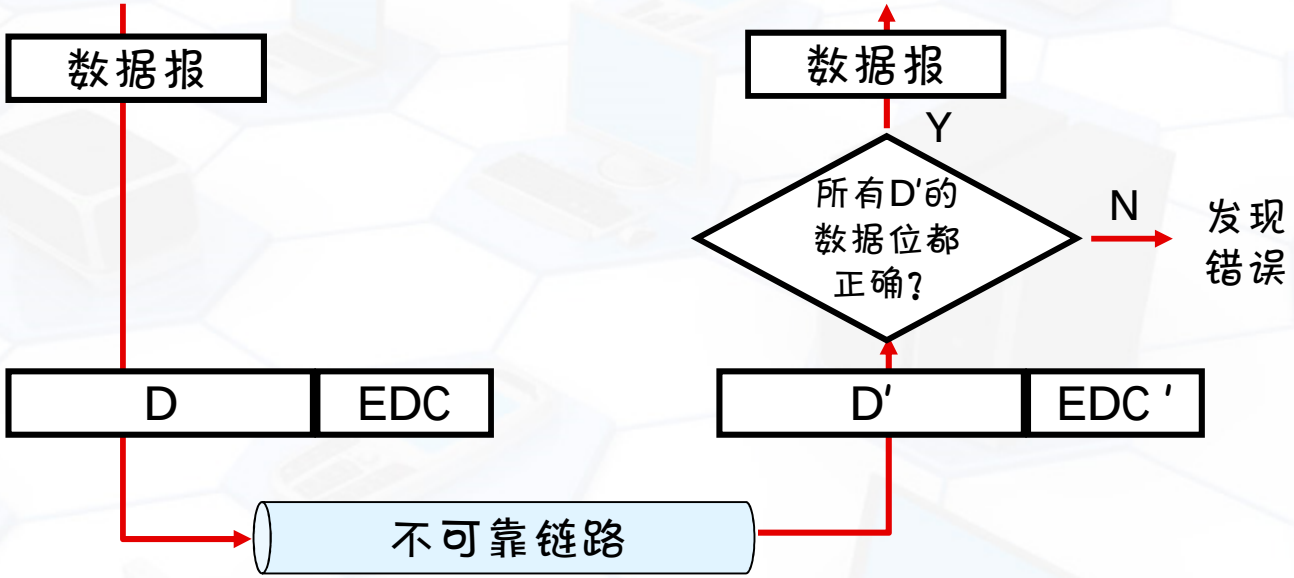
- 只适用于物理层编码有冗余的网络
- 如：802标准的LAN，物理层采用曼彻斯特或差分曼彻斯特编码 用高-低电平对/低-高电平对表示1/0，高-高/低-低电平对不表示数据，可以用来做定界符

三、 差错控制

一般方法：接收方给发送方一个反馈（响应）

- 出错情况
  - 帧（包括发送帧和响应帧）出错；
  - 帧（包括发送帧和响应帧）丢失
- 数据链路层的一个主要功能是通过计时器和序号保证每帧最终交给目的网络层仅一次

- 差错出现的特点：随机，连续突发
- 处理差错的两种基本策略
  - 使用纠错码：发送方在每个数据块中加入足够的冗余信息，使得接收方能够判断接收到的数据是否有错，并能纠正错误
  - 使用检错码：发送方在每个数据块中加入足够的冗余信息，使得接收方能够判断接收到的数据是否有错，但不能判断哪里有错



EDC= Error Detection and Correction bits (冗余数据)

- 检错并非100%可靠，协议工作时也偶尔会遗漏某些错误
- EDC部分越长对检错和纠错越好

1. 纠错码

- 码字（codeword）：一个帧包括m个数据位，r个校验位，n = m + r，则此n比特单元称为n位码字
- 海明距离（Hamming distance）：两个码字之间不同的比特位数目

§例：0000000000 与0000011111的海明距离为5

§为了检测d个错误，需要一个海明距离为d+1的编码方案

Ø这样的编码方案中，d个1位错误不可能将一个有效码字改编成另一个有效码字。当接收方看到一个无效码字的时候，就知道已经发生了传输错误

§为了纠正d个错误，需要一个距离为2d+1的编码方案

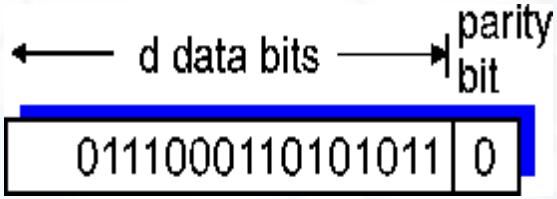
Ø这样的编码方案中，合法码字之间的距离足够远，即使发生了d位变化，则还是原来的码字离它最近，可唯一确定原来的码字，达到纠错的目的

2. 检错码

使用纠错码传数据，效率低，适用于不可能重传的场合；大多数情况采用检错码加重传

a) 奇偶检验

单比特奇偶校验——发现单一二进制位的错误

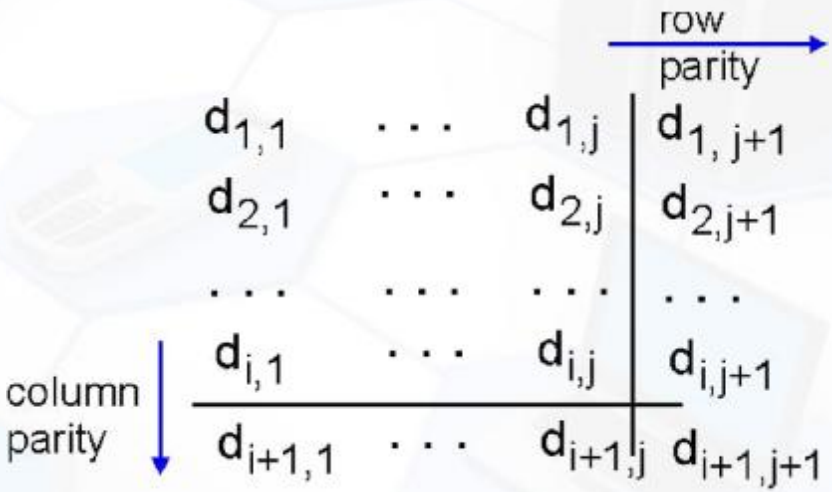


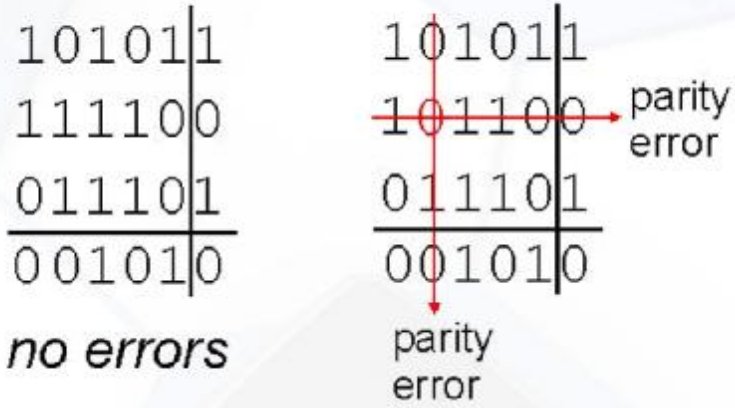
例：使用偶校验（“1”的个数为偶数）

10110101 ——> 101101011

10110001 ——> 101100010

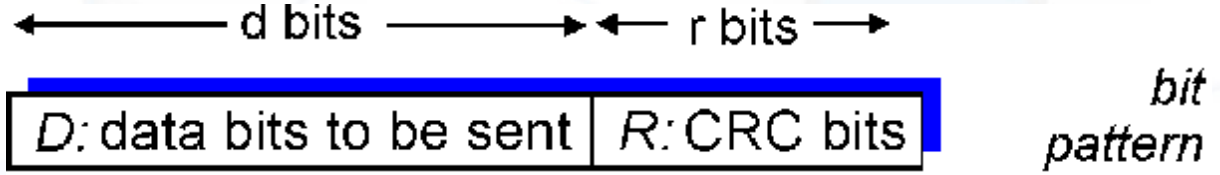
二维奇偶校验——发现并纠正单一二进制位错误





b) 校验和（checksum）:循环冗余校验CRC

- 将数据 D 的各二进制位看做二进制数字
- 选用一个生成多项式 G (有r+1位)
  - 如：110001，可表示成多项式 $x^5 + x^4 + 1$
  - 生成多项式的高位和低位必须为1
- 选择 r个 CRC 位, R



$D * 2^r$ XOR R

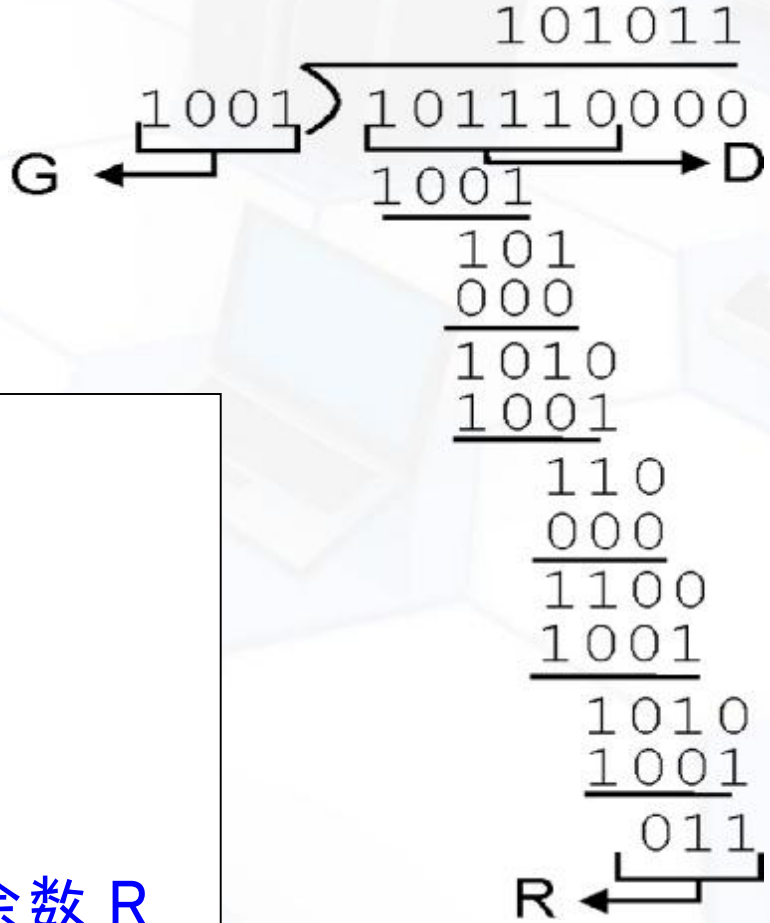
mathematical formula

- 能够检测到所有小于r+1 位的突发错误
- 广泛地用于多种链路级协议 (ATM, HDLC)

§ CRC码基本思想

校验和加在帧尾，使带校验和的帧的多项式能被G(x)除尽；收方接收时，用G(x)去除它，若有余数，则传输出错

$$R = \text{取余数}\left[\frac{D\cdot 2^r}{G}\right]$$



要计算：

$D\cdot 2^r$ XOR R = nG

相当于：

$D\cdot 2^r$ = nG XOR R

即：

$D\cdot 2^r$ 除以 G, 得到余数 R

§ 四个多项式已成为国际标准

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

CRC-16 = $x^{16} + x^{15} + x^2 + 1$

CRC-CCITT = $x^{16} + x^{12} + x^5 + 1$

CRC-32 <略>

§ 硬件完成CRC校验

网卡NIC（Network Interface Card）

四、流量控制与滑动窗口协议（Sliding Window Protocol）

§ 基于反馈机制

§ 流量控制主要在传输层实现

1. 链路层的流量控制策略

1）XON/XOFF方案。使用一对控制字符实现流量控制；发生过载时，接收方向发送方发送一个XOFF字符，使之暂停发送；状况消除后，再向发送方发送一个XON字符，使之恢复发送

2）滑动窗口。利用接收方缓冲区和发送方缓冲区来实现流量控制；通过限制已发送而尚未确认帧的数量，以实现对发送方的控制

2. 滑动窗口协议

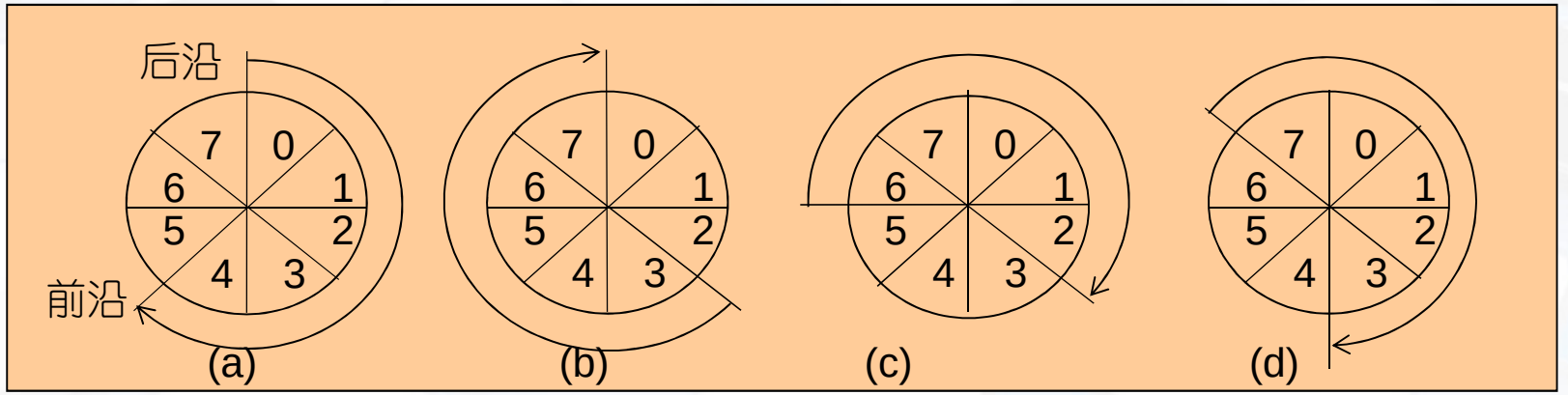
工作原理

- 发送的信息帧都有一个序号，从0到某个最大值，$0 \sim 2^n - 1$，一般用n个二进制位表示
- 发送端始终保持一个已发送但尚未确认的帧的序号表，称为发送窗口。发送窗口的上界表示要发送的下一个帧的序号，下界表示未得到确认的帧的最小编号
- 发送窗口大小 = 上界 - 下界，大小可变

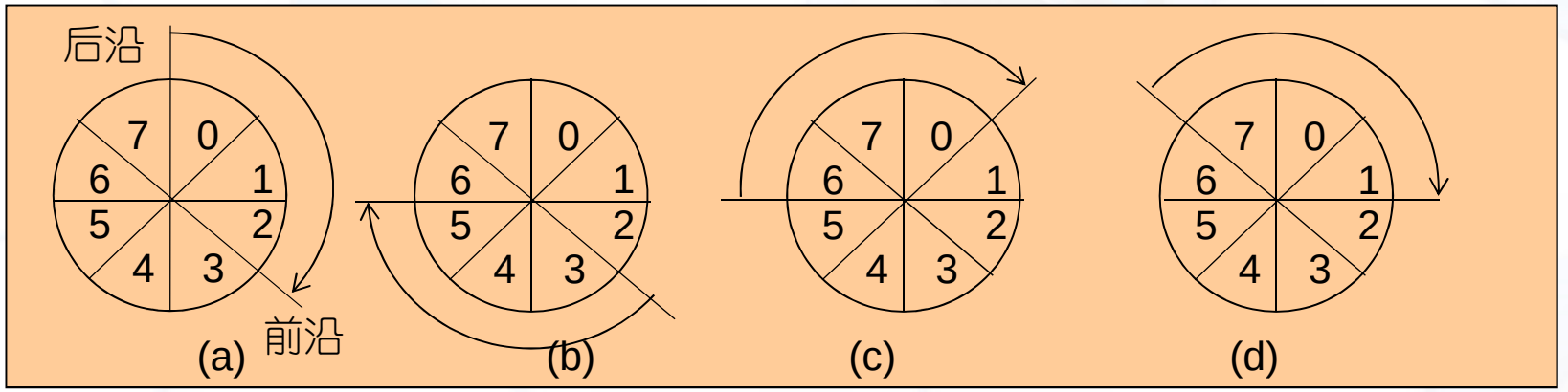
工作原理（续）

- 发送端每发送一个帧，序号取上界值，上界加1；每接收到一个正确响应帧，下界加1
- 接收端有一个接收窗口，大小固定，但不一定与发送窗口相同。接收窗口的上界表示允许接收的序号最大的帧，下界表示希望接收的帧
- 接收窗口容纳允许接收的信息帧，落在窗口外的帧均被丢弃。序号等于下界的帧被正确接收，并产生一个响应帧，上界/下界都加1。接收窗口大小不变

滑动窗口原理示意图（设W_T=5，W_R=3）



发送方



接收方

三个常见窗口协议

- 发送窗口W_T=1，接收窗口W_R=1

  一位滑动窗口协议

- 发送窗口W_T=7，接收窗口W_R=1

  后退n帧的滑动窗口协议

- 发送窗口W_T=4，接收窗口W_R=4

  选择性重发滑动窗口协议

1）停等协议（一比特滑动窗口协议）

- 协议特点
  - 窗口大小：N = 1，发送序号和接收序号的取值范围：0，1
  - 可进行数据双向传输，信息帧中可含有确认信息（piggybacking技术）
  - 信息帧中包括两个序号域：发送序号和接收序号（已经正确收到的帧的序号）
- 存在问题
  - 能保证无差错传输，但是基于停等方式
  - 若双方同时开始发送，则会有一半重复帧
  - 效率低，传输时间长

停等协议的情况1：运行过程正常

A发送（0，1，A0）	Seq ack	A发送A0，seq=0
	B收到（0，1，A0）▲	B收到A0
	B发送（0，0，B0）	B发送B0，并确认A0
A收到（0，0，B0）▲		A收到B0及对A0的确认
A发送（1，0，A1）		A发送A1及对B0的确认
	B收到（1，0，A1）▲	B收到A1及对B0的确认
	B发送（1，1，B1）	B发送B1及对A1的确认
A收到（1，1，B1）▲		A收到B1及对A1的确认
A发送（0，1，A2）		A发送A2及对B1的确认
	B收到（0，1，A2）▲	B收到A2及对B1的确认
	B发送（0，0，B2）	B发送B2及对A2的确认
A收到（0，0，B2）▲		A收到B2及对A2的确认
A发送（1，0，A3）		A发送A3及对B2的确认
	B收到（1，0，A3）▲	B收到A3及对B2的确认
	B发送（1，1，B3）	

停等协议的情况2：运行过程不正常

A发送（0，1，A0）	B发送（0，1，B0）	B也已从网络层得到B0
	B收到（0，1，A0）▲	B以为B0分组A没有收到
	B发送（0，0，B0）	所以B又重复B0分组
A收到（0，1，B0）▲		A以为A0分组B没有收到
A发送（0，0，A0）		所以A又重复A0分组
	B收到（0，0，A0）▲	B收到A0及对B0的确认
	B发送（1，0，B1）	B发送B1及对A0的确认
A收到（0，0，B0）▲		A收到B0及对A0的确认
A发送（1，0，A1）		A发送A1及对B0的确认
	B收到（1，0，A1）▲	B以为B1分组A没有收到
	B发送（1，1，B1）	所以B又重复B1分组
A收到（1，0，B1）▲		A以为A1分组B没有收到
A发送（1，1，A1）		所以A又重复A1分组
	B收到（1，1，A1）▲	
	B发送（0，1，B2）	

2） 连续ARQ协议

- 为提高传输效率而设计
- 停等协议的主要问题是信道利用率太低。发送端的等待时间至少是发送端到接收端传播时间的二倍
- 信道的利用率为：**发送时间/来回时间**
- 管道化（pipelining）：在等待确认的时间内继续发送
- 存在的问题：若信道不可靠，使发送端在意识到数据丢失时，已有大量数据到达接收端

例：卫星信道传输速率50kbps，往返传输延迟500ms，若传1000bit的帧，若使用停等协议，则传输一个帧所需时间为：

发送时间 + 数据传输延迟 + 确认传输延迟（确认帧很短，可忽略发送时间）= 1000bit / 50kbps + 250ms + 250ms = 520ms

信道利用率 = 20 / 520 » 4%

Ø 对一般情况而言

若信道带宽*b*比特/秒，帧长度*L*比特，往返传输延迟*R*秒，则信道利用率为

$$(L/b) / (L/b + R) = L / (L + Rb)$$

Ø 结论：传输延迟大，信道带宽高，帧短时，信道利用率低

- 解决效率的办法
  - 连续发送多帧后再等待确认，称为流水线技术
  - 带来的问题：信道误码率高时，对坏帧和非坏帧的重传非常多
- 两种常见的连续ARQ方法
  - 退后n帧（go back n）

    接收方从出错帧起丢弃所有后继帧，接收窗口为1。对于出错率较高的信道，浪费带宽
  - 选择重传（selective repeat）

    接收窗口大于1，先暂存出错帧的后继帧，只重传坏帧，对最高序号的帧进行确认；接收窗口较大时，需较大缓冲区

a) 退后n帧协议

Ø发送方有流量控制，为重传设缓冲区

Ø发送窗口大小＜序号个数

如果窗口大小与序号个数一致（如序号个数为0～7），则会有下面情况发生：

1) 发送方发送帧 0～7

2) 序号为 7 的帧的确认被捎带回发送方

3) 发送方发送另外 8 个帧，序号为 0～7

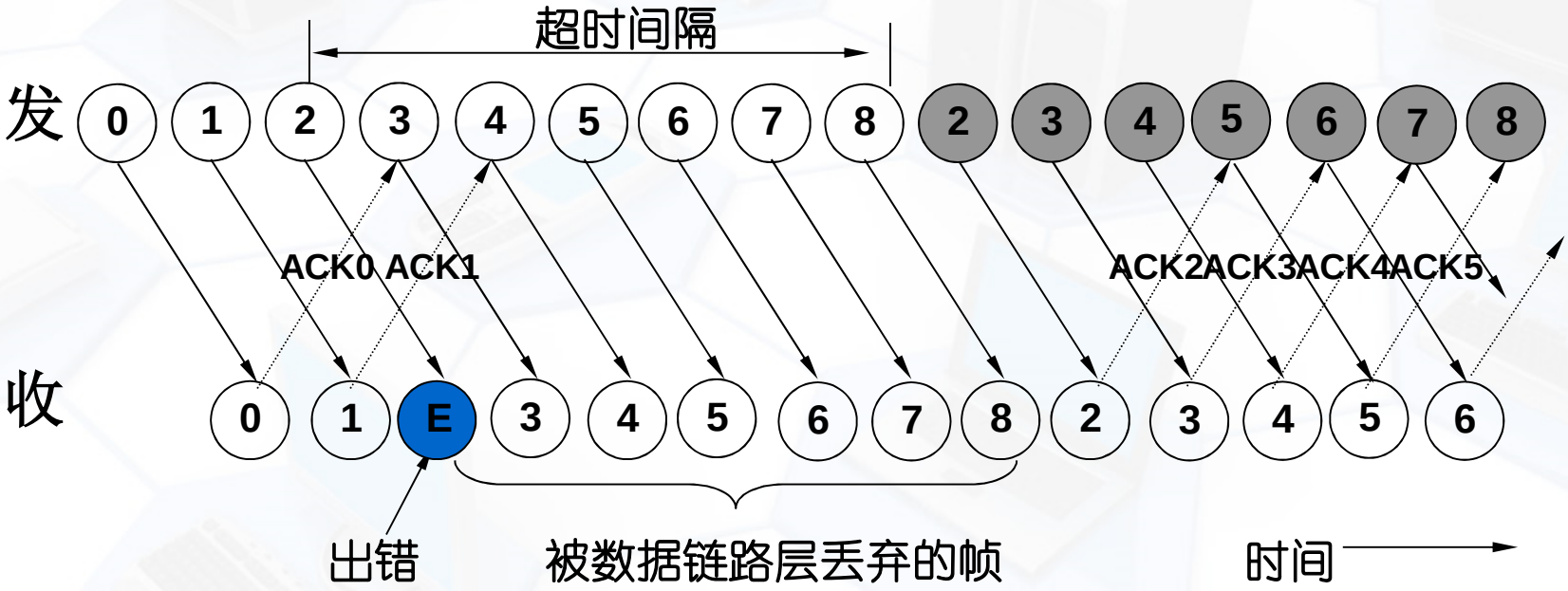
4) 另一个对帧 7 的捎带确认返回

问题是：第二次发送的 8 个帧成功了还是丢失了？

Ø当出现差错时退后n帧重发

Ø由于有多个未确认帧，设多个计时器

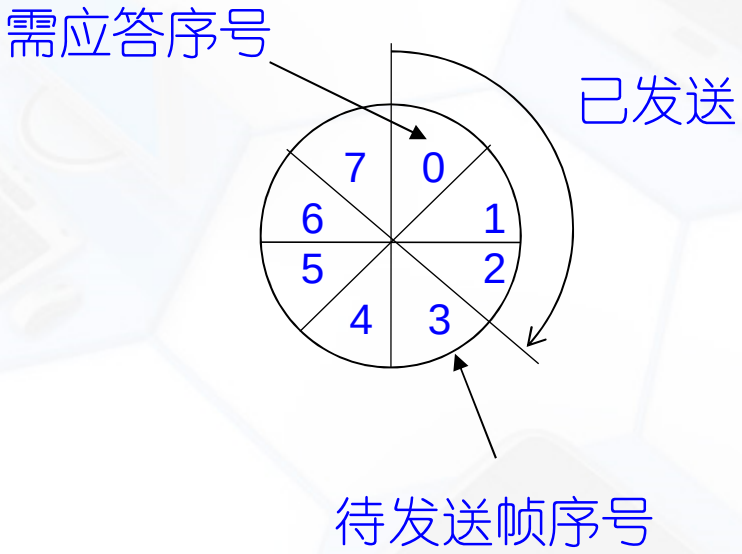
后退n帧的滑动窗口协议图例



有一个差错时后退n帧（发送窗口为7）

后退n帧的窗口定义

- 如果编号由N位组成，则发送窗口$W_T=2^n-1$，接收窗口$W_R=1$
- 即管道化（pipelining）协议
- 对于3位二进制的序号，则（$W_T=7$，,$W_R=1$）



b) 选择重传协议

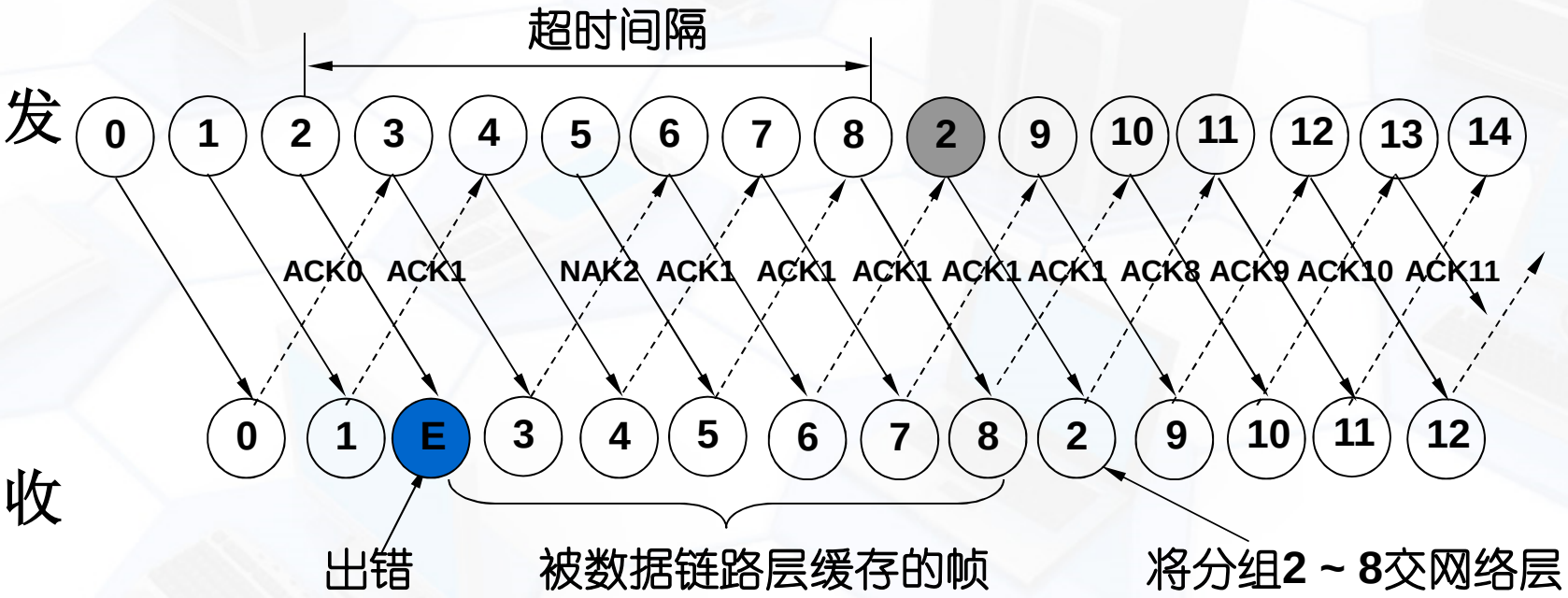
在不可靠信道上有效传输时，不会因重传而浪费信道资源，有选择的重传

发送窗口大小：按最大序号，接收窗口大小：序号个数的一半

- 需保证接收窗口前移后与原窗口没有重叠
- 缓冲区设置

  发送方和接收方的缓冲区大小应等于各自窗口大小
- 增加确认计时器，解决两个方向负载不平衡带来的阻塞问题
- 可随时发送否定性确认帧NAK

选择性重传窗口协议图例



有一个差错时仅重发一帧（接收窗口和发送窗口都为7）

选择性重传协议失败的情况

考虑一种情况：序号为3位二进制，即帧号为0 ~ 7
发送窗口和接收窗口的大小均为7

- 发送方连续发送了7帧，帧号为0 1 2 3 4 5 6，然后等待确认
- 在尚未接收到帧前，接收窗口为0 1 2 3 4 5 6。发送方发送的7帧正确地收到后，接收方发出了确认6，意即0 ~ 6帧全部收到，然后取出分组交网络层、清缓冲区并调整窗口为7 0 1 2 3 4 5
- 发送方一直在等待确认，但确认帧由于某种原因失踪了，在超时后，发送方又重发0 1 2 3 4 5 6帧，并等待确认

- 接收方收到0 1 2 3 4 5 6帧，认为是第二批来的帧，照正常处理，发现0 1 2 3 4 5均在其接收窗口内，当然接收并存入缓冲，6丢弃，但由于应首先到达的7未到，所以只能发6 ack，意即再次确认上次收到的0～6
- 但在发送方来看，收到了6 ack后才知道，重发的0～6总算收到了，于是，调整窗口为7 0 1 2 3 4 5，从网络层取分组，并发送第二批帧
- 接收方在收到7 0 1 2 3 4 5后，发现0 1 2 3 4 5帧已在缓存中，是重复的，应丢弃，7接收。然后交网络层，清缓冲区、调整接收窗口
- 此时，接收方的网络层发现：数据链路层交来的第二批分组中的0 1 2 3 4 5与原来的重复

选择性重传协议
当W_T=W_R=7失败的原因

- 原因在于接收窗口过大，新窗口与原窗口中的有效顺序号有重叠
- 所以，发送窗口+ 接收窗口 = 2^n；

  且：发送窗口 >= 接收窗口

选择性重传协议正常工作的情况

- 发送窗口 = 接收窗口 = (MAX_SEQ + 1) / 2
- 增加了否定性确认NAK，当收到一个坏帧，或收到一个非期望的帧，则发一个NAK帧
- 增加了一个辅助定时器，当收到一个正确的帧，而没有可捎带确认的数据帧，当辅助定时器超时，则立即发送一个ACK帧

五、常用的数据链路层协议

数据链路层协议分类

- 面向字符的链路层协议
  - ISO的IS1745，基本型传输控制规程及其扩充部分（BM和XBM）
  - IBM的二进制同步通信规程（BSC）
  - DEC的数字数据通信报文协议（DDCMP）
  - PPP

- 面向比特的链路层协议
  - IBM的SNA使用的数据链路协议SDLC（Synchronous Data Link Control protocol）
  - ANSI修改SDLC，提出ADCCP（Advanced Data Communication Control Procedure）
  - ISO修改SDLC，提出HDLC（High-level Data Link Control）
  - CCITT修改HDLC，提出LAP（Link Access Procedure）作为X.25网络接口标准的一部分，后来改为LAPB

1. 高级数据链路控制规程HDLC

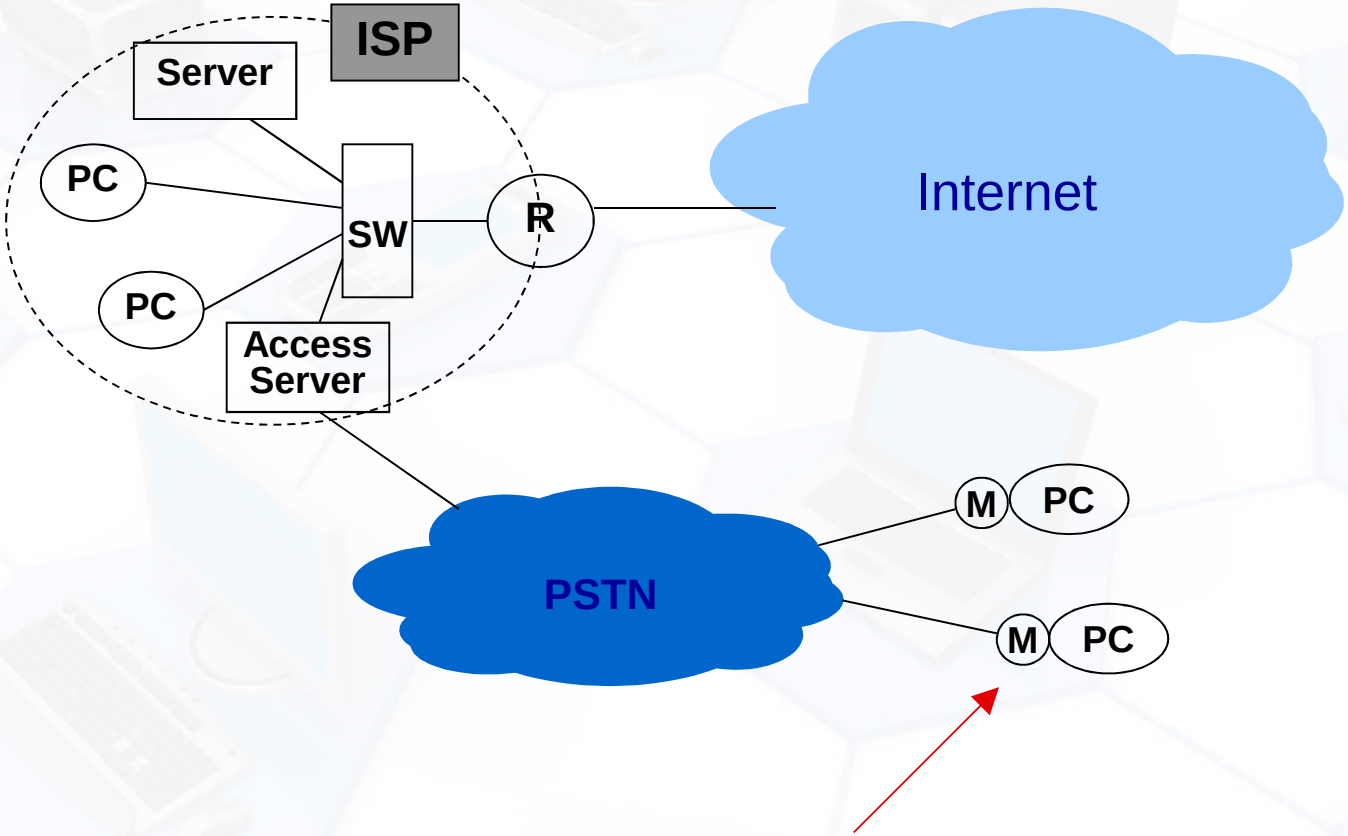
- 1976年，ISO提出HDLC（High-level Data Link Control）
- HDLC的组成
  - 帧结构
  - 规程元素
  - 规程类型
  - 使用HDLC的语法可以定义多种具有不同操作特点的链路层协议
- HDLC的适用范围
  - 计算机 —— 计算机
  - 计算机 —— 终端
  - 终端 —— 终端

HDLC的帧格式

8	8	8	>=0	16	8
01111110	地址	控制	数据	校验和	01111110

帧标志序列	即01111110，作为帧的分隔标志，如线路空闲，则用标志序列填充，用位插入方法实现透明传输
地址域	在总线型多终端情况下，是终端的站号；在点对点的情况下，用来标志命令和响应
控制域	定义帧的类型、序号等和其它一些功能
数据域	用户数据，长度任意
校验和	CRC码，从地址字段到信息字段

2. 点对点协议PPP



远程家庭用户通过拨号与ISP联接

- PPP是Internet标准（RFC1661, 1662, 1663）
  - 处理错误监测
  - 支持多种协议（IP、IPX、DECnet等）
  - 连接时允许协商IP地址，允许动态分配
  - 允许身份认证
  - 提供差错校验
  - 以帧为单位发送，而不是原始IP包

- 点到点协议 PPP 包括两部分
  - 链路控制协议LCP（Link Control Protocol）

    可使用多种物理层服务：modem，HDLC串线，SDH/SONET等
  - 网络控制协议NCP（Network Control Protocol）

    可支持多种网络层协议

- 帧格式与HDLC相似，区别在于PPP是面向字符的，采用字符填充技术

p PPP的帧格式

PPP的帧格式类似于HDLC，但是面向字符的协议（以字节为单位）

1	1	1	1/2	可变	2/4	1
标志 01111110	地址 11111111	控制 00000011	协议	有效载荷	校验和	标志 01111110

标志域：固定为01111110，与HDLC相同

地址域：固定为11111111

控制域：缺省为00000011，即无序号帧（即毋需确认）

协议域：不同的协议不同的代码

载荷域：可变长，缺省最长1500字节

校验和：缺省为2字节，也可为4字节，仅是头部的校验和

p PPP的工作过程

- ü 发送端PPP首先发送LCP帧，以配置和测试数据链路
- ü 在LCP建立好数据链路并协调好所选设备之后，发送端PPP发送NCP帧，以选择和配置一个或多个网络协议
- ü 当所选的网络层协议配置好后，便可将各网络层协议的分组发送到数据链路上
- ü 配置好的链路将一直保持通信状态，直到LCP帧或NCP帧明确提示关闭链路，或有其它的外部事件发生（如用户干预等）

一次使用PPP协议的过程

①初始状态

②建立连接：建立成功到③，否则到①

③选项协商：协商成功到④，否则到⑦

④身份认证：认证成功到⑤，否则到⑦

⑤配置网络：网络配置完后到⑥

⑥数据传输：数据传输完后到⑦

⑦释放链路：回到①

六、多路访问链路和协议

两种不同类型的链路

- 点对点
  - 拨号访问使用的PPP协议
  - 以太网中主机和交换机间的点对点连接
- 广播 (共享介质)
  - 传统以太网
  - 无线局域网

1. 多路访问协议

- 共享的单一广播信道
- 两个以上的节点同时发送数据产生干扰——冲突
  - ü冲突：节点同时收到两个以上的信号
- 多路访问协议利用算法决定哪个节点能够使用信道发送数据
- 协议的通讯也通过共享信道自身，不存在额外的控制信道

2. 理想的多路访问协议

- 如果一个节点需要发送，它可以使用全部带宽
- 如果有M个节点要发送，每个节点平均使用带宽的M分之一
- 全分散型的信道访问:
  - ü无专用节点协调、管理数据发送
  - ü无同步时钟、时隙等
- 协议简单

3. MAC协议

- 信道分割方式
  - 将信道分为较小的“片段”(按时隙，频率，编码等)
  - 每个片段供各节点单一使用
- 随机访问方式
  - 不划分信道，允许冲突
  - 发生冲突能够恢复到正常传送
- 轮询方式
  - 每个节点轮流发送，数据较多的节点要等较长时间

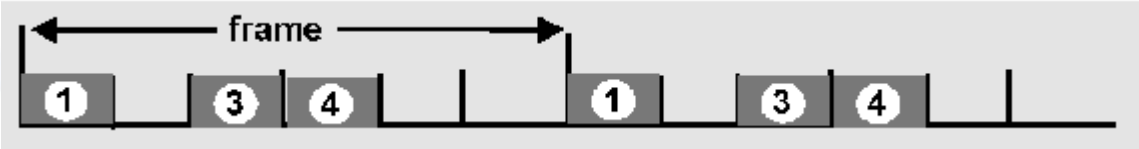
u 信道分割MAC协议

TDMA: 时分多路访问

§ 每个站有固定时长的时隙

§ 不用的时隙为空闲

§ 如: 有6个站点的LAN, 1,3,4 有数据发送, 时隙2,5,6 空闲

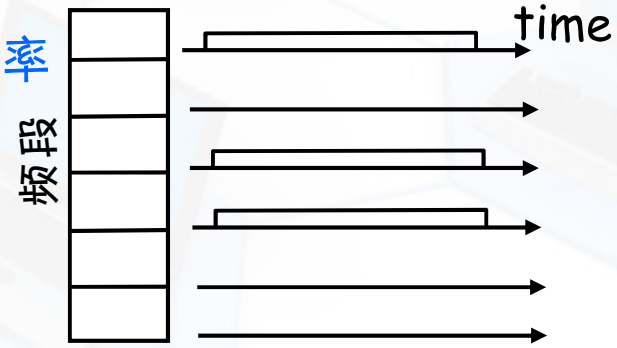


FDMA: 频分多路访问

§ 信道按频段进行划分，每个站点使用固定频率

§ 不发送数据时频段空闲

§ 如: 有6个站点的LAN, 1,3,4在发送数据, 2,5,6频段空闲



CDMA: 码分多路访问

§ 每个站点有唯一的 “代码”，所有用户共享相同的频率，但每个用户有自己的码片序列进行编码

§ 编码数据 = (原始数据) X (码片序列)

§ 解码: 编码信息与码片序列的内积

§ 允许多个用户共存并同时发送数据，且干扰最少闲