XI'AN JIAOTONG UNIVERSITY

数据通信与计算机网络

第3章 数据链路层

主要内容与基本要求

- **主要内容**
  - 数据链路层的基本概念；
  - 停止等待协议、连续ARQ协议的工作原理及性能分析；
  - 面向比特的链路层规程HDLC和因特网的点对点协议PPP。
- **基本要求**
  - 理解数据链路层的基本概念、主要功能；
  - 掌握停止等待协议及其性能分析；掌握传输时延、吞吐量和信道利用率等概念；
  - 掌握连续ARQ协议的工作原理及性能分析，理解滑动窗口的概念；
  - 了解选择重传ARQ的基本思想；
  - 掌握HDLC协议的主要内容、帧格式。了解PPP协议。

本章目录

3.1 数据链路层的基本概念

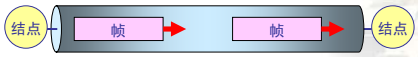
任务：在相邻结点间的不可靠的物理链路上传送以帧为单位的数据。

数据链路层的许多概念都是计算机网络的重要概念，其中的部分原理对其它层次的协议也是适用的。

链路（link）就是一条**无源的点到点的物理线路段**，中间没有任何其它交换结点。又称**物理链路**。是物理概念。

数据链路（data link），当传输数据时，除必须有一条物理线路外，还需要一些必要的通信协议来控制这些数据传输。**实现这些协议的硬件和软件加上链路就构成了数据链路**。又称**逻辑链路**，是逻辑概念。

在讨论数据链路层功能时，常把两个对等的数据链路层之间的通信看成数据帧通过一条**数字管道**。“帧”是数据链路层传输数据的单位。



3.1 数据链路层的基本概念

历史上曾把通信协议叫做通信**规程**（procedure），是协议的同义语。

数据链路层的主要功能如下（有些协议可以省略一些）：

（1）**链路管理**，数据链路的建立、维持和释放。

（2）**帧定界**，收方能从收到的比特流中准确地区分一帧的开始和结束在什么地方。又称**帧同步**。

（3）**流量控制**，发方发送数据的速率必须使收方来得及接收。

（4）**差错控制**，用于检测或纠正帧传输过程中出现的差错的机制。

（5）**将数据和控制信息分开**，能区分出同一帧中的数据和控制信息。

（6）**透明传输**，不管什么样的数据组合都能在链路上传送。

（7）**寻址**，保证每一帧都能送到正确的目的站。当然，收方也应知道发方是那个站。

本章实际上讨论的是最基本的数据链路层协议。

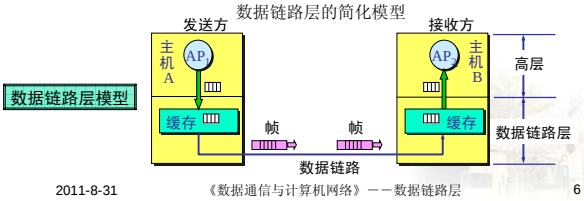
3.2 停止等待协议

停止等待协议是最简单、最基本的数据链路层协议。

链路层使用物理层提供的不可靠（差错、丢失等）的服务。数据链路层协议主要考虑两个问题：**避免数据出现差错和丢失；发送速率要适应收方的接收能力。**

3.2.1 完全理想化的数据传输

数据链路层的简化模型



3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

假设数据传输符合以下两个条件：

假设1：链路是理想的传输信道，所传送的数据既不出错也不丢失。

假设2：不管发送速率多快，收方总能收下并及时上交主机。相当于认为接收端向主机交付数据的速率永远不会低于发送端的发送速率。

在这样的理想化条件下，数据链路层就既不需要差错控制，也不需要流量控制。

3.2.2 具有简单流量控制的数据链路层协议

现在**去掉假设2，保留假设1**。那么就需要考虑流量控制问题。流量控制协议使发送端发送数据的速率能适应接收端的接收能力。

最直接的想法就是由接收方通过某种机制控制发送方的发送速率。这种由收方控制发方的数据流，是计算机网络中流量控制的一个基本方法。

3.2 停止等待协议

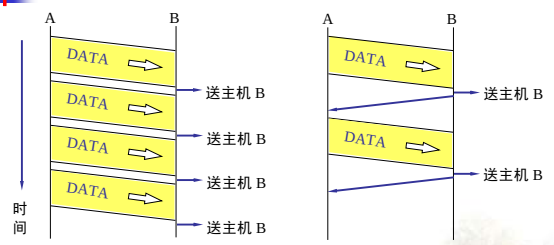
XI'AN JIAOTONG UNIVERSITY

在发送结点	在接收结点
（1）从主机取一个数据帧。 （2）将数据帧送到数据链路层的发送缓存。 （3）将发送缓存中的数据帧发送出去。 （4）等待。 （5）若收到由接收结点发过来的信息[1]，清空缓存，转（1）。	（1）等待。 （2）若收到由发送结点发过来的数据帧，则将其放入数据链路层的接收缓存。 （3）将接收缓存中的数据帧上交主机。 （4）向发送结点发一信息[1]，表示数据帧已经上交主机。 （5）转（1）。

[1]：接收方向发送方发送信息的格式与内容由双方事先商定。由于数据在传输过程中不会出错，因此这个信息不需要有任何具体内容，称为**哑元帧**（dumb）。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY



不需要数据链路层协议的数据传输（左图）和具有简单流量控制的数据链路层协议（右图）

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

3.2.3 实用的停止等待协议

去掉两个假设，考虑实际情况。有一个前提，接收方能检测出收到的帧是否出错。

（1）若**数据帧不出错**，则收方发**确认帧**ACK（ACKnowledgement），发方收到ACK后发下一帧。

（2）若收到的**数据帧出现了差错**，收方发**否认帧**NAK（Negative ACK），发方收到后重传数据帧，直到收到ACK为止。

那么，只有以上两条可以吗？答案是否定的。因为，协议需要考虑到所有不利的情况。假设由于某种原因（如瞬间强烈的干扰），数据帧或应答帧（ACK或NAK）丢失，那么双方就会**永远等下去**，即出现**死锁**。因此，协议必须解决这个问题。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

（3）**帧丢失**，为了防止双方一直等下去，发方在发送完一个数据帧后启动一个**超时计时器**（timeout timer）。若到了超时计时器所设定的时间发方还没有收到应答帧，则重传数据帧。**重传时间一般可选为略大于"从发完数据帧到收到确认帧所需的平均时间"**。

问题完全解决了吗？No！（3）只是解决了死锁问题。现在假设应答帧丢失，那么由（3），发方会重传数据帧，如此收方会收到两个同样的数据帧。这是不允许的。

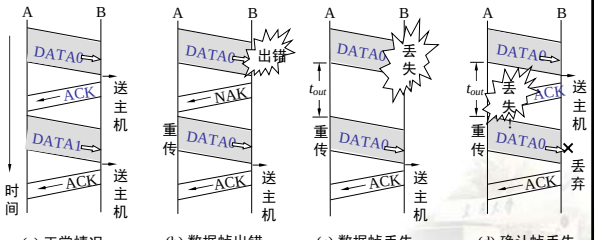
（4）**重复帧**，要解决重复帧问题，必须让每个数据帧带上不同的发送序号。收方根据数据帧的序号识别并丢弃重复帧。要注意的是，**收方丢弃重复帧后还必须发一个确认帧ACK**。

序号所占用的位数一定是有限的，要重复使用。位数越大开销越大，对停止等待，用一个比特编号就够了。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

从以上讨论可以看出，虽然物理层不可靠，但由于停止等待协议采用了有效的检错重传机制，使数据链路层对上面的网络层提供了可靠的点对点传输服务。



(a) 正常情况 (b) 数据帧出错 (c) 数据帧丢失 (d) 确认帧丢失

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

3.2.4 循环冗余校验的原理

前面提到，接收方要检测收到的数据帧是否有错误，这就要用到差错检测技术。

常见的差错检测技术有奇偶校验、循环冗余校验（CRC）等。

为了检测数据帧是否出现差错而在数据后面添加上的冗余码常称为**帧校验序列FCS**（Frame Check Sequence）。它的作用是要保证收到的数据和发送的数据完全相同。

仅用CRC差错检测技术只能做到**无差错接受**（accept），即**“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”**。注意接受和接收的区别。

要做到**“可靠传输”**必须加上确认和重传机制。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

3.2.5 停止等待协议的算法

在发送结点

(1) V(S) ← 0。 {发送状态变量初始化}
(2)从主机取一个数据帧。
(3) N(S) ← V(S)。 {将发送状态变量的值写入发送序号}
将数据帧送交发送缓存。
(4) 将发送缓存中的数据帧发送出去。
(5) 设置超时定时器。 {选择合适的重传时间}
(6) 等待。 {等待以下(7)~(9)三个事件中最先出现的一个}
(7) 若收到ACK，则从主机取一个新的数据帧；
V(S) ← [1-V(S)]。 {更新状态变量，变为下一序号}，转(3)。
(8) 若收到NAK，转(4)。
(9) 若超时计时器时间到，转(4)。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

在接收结点

(1) V(R) ← 0。 {接收状态变量初始化}
(2) 等待。
(3) 当收到一个数据帧，就检查是否出错。
若检查结果正确无误，则执行后续算法；
否则转(8)。
(4) 若N(S)=V(R)，则执行后续算法； {收到序号正确的帧}
否则丢弃此数据帧，然后转(7)。
(5) 将收到的数据帧中的数据部分送交主机。
(6) V(R) ← [1-V(R)]。 {更新接收状态变量}
(7) 发送确认帧ACK，转(2)。
(8) 发送否认帧NAK，转(2)。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

需要注意如下几点：

（1）发送数据帧时，要把发送状态变量V(S)的值写入数据帧的发送序号N(S)上。只有收到确认帧时，才更新V(S)并发送新的数据帧。

（2）每收到一个数据帧， 把数据帧中的序号N(S)与接收状态变量V(R)比较，若相等表明是新帧，否则为重复帧。

（3）若为重复帧，丢弃，保持接收状态向量不变并发确认帧。

（4）上面的算法在数据帧出错时发NAK，实际中常用的是若数据帧出错，收方丢弃后什么也不做，等待发方的超时重传。

（5）发方发完数据后，要在缓存中保留数据帧的副本以便出错重传，只有在收到ACK后，才清除此副本。

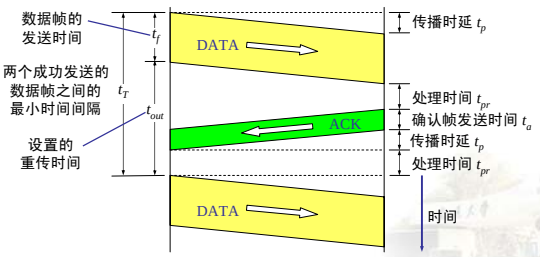
由于发送端对出错的数据帧进行重传是自动进行的，所以把这种差错控制机制称为ARQ（Automatic Repeat reQuest），即**自动请求重传**。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

3.2.6 停止等待协议的定量分析

针对**半双工通信模型**对停止等待协议的性能进行定量分析。



3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

由图可知，重传时间可以设为

$$t_{out} = t_p + t_{pr} + t_a + t_p + t_{pr} \quad (3\text{-}2)$$

如果设帧处理时间和确认帧的发送时间均远小于传播时延，则重传时间可以简单地取为两倍的传播时延，即

$$t_{out} = 2t_p \quad (3\text{-}3)$$

假设信道不出错，那么成功发送一个数据帧的最小时间为

$$t_T = t_f + t_{out} = t_f + 2t_p \quad (3\text{-}4)$$

现在假设数据帧出现差错（包括丢失）的概率为 p，且假设应答帧不出错，重传次数不受限制。在此基础上可以推导正确传送一个数据帧所需的平均时间 t_{av}，显然，它等于成功传输一个帧所需的平均次数 N 乘上发送一帧占用的信道时间 t_T。

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

根据前面的假设，并利用离散随机变量的数学期望公式，可得：

$$N=\sum_{i=1}^{\infty} ip^{i-1}(1-p)=1/(1-p)$$

因此，

$$t_{av}=Nt_T=t_T/(1-p) \quad (3\text{-}5)$$

单位是**秒每帧**。

通常把每秒成功发送的最大帧数定义为链路得最大吞吐量 λ_{max}：

$$\lambda_{max}=1/t_{av}=(1-p)/t_T \quad (3\text{-}6)$$

单位是**帧每秒**。

设发送端数据帧的实际到达率为 λ，则

$$\lambda \le (1-p)/t_T \quad (3\text{-}7)$$

3.2 停止等待协议

XI'AN JIAOTONG UNIVERSITY

为了更公平地比较不同协议的性能，常使用**归一化吞吐量**，定义为在帧的发送时间内成功发送的平均帧数。即用帧的发送时间进行归一化。

$$\rho \equiv \lambda t_f \le (1-p)/\alpha < 1 \quad (3\text{-}8)$$

其中 α 是 t_T 的归一化时间：

$$\alpha \equiv t_T/t_f \ge 1 \quad (3\text{-}9)$$

停止等待协议的优点是比较简单，但缺点是通信**信道的利用率不高**。为了克服这个缺点，产生了连续ARQ和选择重传ARQ。

书上（3-5）式的解释

$$t_{av}=t_T+t_T\sum_{i=1}^{\infty} ip^i(1-p)=t_T/(1-p)$$

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

3.3.1 连续ARQ协议的工作原理

停止等待协议信道利用率低的主要原因是等待时间太长，要想办法减小它。

连续ARQ的**基本思想**：如果**信道出错的概率很小**，那么发送数据帧成功的概率很大。所以在收到应答帧前，我们可以假设收方能够正确的收到所发送的数据帧（这个概率很大）。在这个假设前提下，我们没有必要等待确认帧，而可以再连续发送若干帧。如果这时收到了确认帧，那么还可以接着发送数据帧。从而减少了等待时间，提高了吞吐量。

连续ARQ协议**付出的代价**是：发送方的缓存要加大；确认帧编号的位数要加大；发送方要同时维护多个超时计时器，管理开销加大。

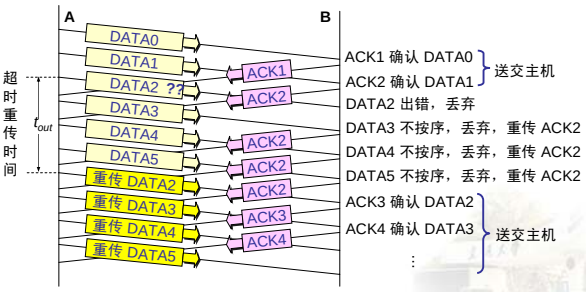
接收方与停止等待协议相同。若数据帧出错，收方有两种处理办法，发送否认帧或不作任何响应。实际中常用后者。

强调一下，连续ARQ的假设前提，信道出错的概率很小。

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

连续ARQ的工作原理



3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

注意以下几点：

（1）**收方只按顺序接收数据帧**。收到不按序的帧，丢弃并且发送已发送过的最后一个确认帧。

（2）ACKn表示收到了n-1号及其前面的所有帧，准备好接收n号帧。

（3）发送方每发完一个数据帧都要设置该帧的超时计时器。如果超时时间到了，仍未收到对应的确认帧，则重传该数据帧并重新设置超时计时器。

注意连续ARQ在重传时的处理，假设2号帧超时时间到没有收到确认，此时不仅要重传2号帧，而且要重传在等待2号帧时已经发送的后续数据帧，如3～5帧，并重设超时计时器。因此被称为Go-back-N ARQ，即回退N ARQ，意思是当出错时，要向回走N个帧。

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

3.3.2 连续ARQ协议的吞吐量

由工作原理图可以看到，信道不出错时，成功发送一帧的时间是 t_f，而当出错时，重传一个数据帧所需的时间为 t_T。参照(3-5)式，可以得到连续ARQ协议正确传送一个数据帧所需的平均时间为

$$t_{av}=t_f+t_T\sum_{i=1}^{\infty} ip^i(1-p)=t_f[1+(\alpha-1)p]/(1-p) \quad (3\text{-}11)$$

其中，参数 α 同（3-9）式，t_T 略大于 t_f+t_{out}。

在发送结点处于饱和状态下，连续ARQ的最大吞吐量为

$$\lambda_{max}=1/t_{av}=(1-p)/\{t_f[1+(\alpha-1)p]\} \quad (3\text{-}12)$$

归一化的吞吐量为

$$\rho=\lambda t_f \le (1-p)/[1+(\alpha-1)p] \quad (3\text{-}13)$$

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

停止等待协议和连续ARQ协议的吞吐量比较，我们把（3-8）式和（3-13）式放在一起。

$$\rho \equiv \lambda t_f \le (1-p)/\alpha < 1$$

$$\rho = \lambda t_f \le (1-p)/[1+(\alpha-1)p]$$

可以看出，若$\alpha \to 1$，则二者均趋于$(1-p)$。

随着α的增大，连续ARQ的性能越来越好。

例

数据帧的差错率为$p=0.01$，参数$\alpha=4$，那么

对停止等待协议$\rho \le 0.99/4$，

而对连续ARQ协议$\rho \le 0.96$。

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

3.3.3 滑动窗口的概念

前面曾提到，为了实现连续ARQ需要对数据帧编号，那么编号应该取多少位呢？显然这取决于发送方在没有收到确认帧的情况下可以连续发送多少帧。这个帧数并不是越大越好，因为当出错时需要重传的帧的数目也会随之增大，对缓存的大小要求也更高。

因此，**应当对已发送出去但未被确认的数据帧的数目加以限制**。当然，这需要加入适当的控制机制。

滑动窗口通过在发送端和接收端分别设定**发送窗口**和**接收窗口**达到此目的。

发送窗口用来对发送端进行流量控制。发送窗口的大小W_T是未收到确认信息的情况下发送方最多可以发送的数据帧的个数。

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

接收窗口控制哪些数据帧可以接收，哪些不能接收。即，**在接收端只有当收到的数据帧的发送序号落入接收窗口内时才允许接收该数据帧，否则丢弃**。用W_R表示接收窗口的大小。

窗口滑动过程

发送窗口规则

（1）窗口内的帧是允许发送的帧，不考虑是否收到确认。

（2）每发送一帧，允许发送帧数减1。发送窗口位置不变。

（3）允许发送的帧发完，还没收到确认，停止发送。

（4）每收到一个确认，发送窗口向前滑动一个位置。

连续ARQ协议的接收窗口规则

（1）只有当收到的数据帧的序号与接收窗口一致时才能接收该帧。否则，丢弃它。

（2）每收到一个序号正确的帧，接收窗口向前滑动一个帧的位置。同时向发送端发送对该帧的确认。

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

不难看出，由于如果发送端没有收到确认，发送窗口就不能移动，所以只有在接收窗口向前滑动时（同时也发了确认），发送窗口才能向前滑动。因此称之为**滑动窗口协议**。

当发送窗口和接收窗口都等于1时，就是停止等待协议。

窗口的最大值

对连续ARQ协议，当发送序号所占的比特数一定时，发送窗口的最大值是多少呢？如用3比特可以编出8个号，发送窗口可以选8吗？No!

设发送窗口大小为8，发送端已发完窗口内的8个帧，并停止发送。假定接收端正确收到了所有8个帧并发送了确认帧。考虑两种如下情况：

（1）**所有确认帧到达发送端**，发送窗口移动；并发送新的帧。

（2）**所有确认帧都丢失了**，发送窗口不动。超时时间到后重传旧的帧。

3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

现在假设接收端收到了“0号帧”，那么它是“新的0号帧”还是“旧的0号帧”？显然，在此情况下，接收端无法区分这两种情况。

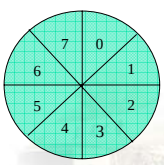
对连续ARQ协议，可以证明：当使用n比特编号时，只有发送窗口的大小$W_T \le 2^n - 1$时，协议才能正确运行。

所有已发送的数据帧用一个先进先出队列保存。

滑动窗口的另一种表示方法如右下图所示。

实际上，发送窗口的大小还应受到一个限制，因为发送窗口与帧的发送时间的乘积大于超时时间就没有实际意义了。所以

$$W_T \le \lfloor t_{out}/t_f \rfloor$$



3.3 连续ARQ协议

XI'AN JIAOTONG UNIVERSITY

3.3.4 信道利用率与最佳帧长

信道利用率是扣除全部控制信息后的数据率与信道容量之比。

控制信息是必须有的，因此信道利用率不可能达到100%。

若帧长取得太短，那么控制信息占的比例就增大，即开销增大，从而导致信道利用率下降。

若帧长选的太长，数据帧出错的概率就增大，重传次数也相应地增大，也会导致信道利用率的下降。

由此可见，一定存在一个最佳帧长，在此帧长下的信道利用率最高。

对于卫星信道，每个比特出错可以视为独立的。

设误比特率为p_b，设数据帧长为l_f，其中控制部分长l_h，数据部分长l_d则误帧率为

$$p = 1-(1-p_b)^{l_f} \approx l_f p_b \ll 1$$

3.3 连续ARQ协议

设发送端工作在饱和状态下，其发送速率为λ_{max}。则可求出平均有效数据率为

$$D=\lambda_{max}l_d=(1-p)l_d\Big/\{t_f[1+(\alpha-1)p]\} \quad (3\text{-}16)$$

设信道容量为 C，显然

$$t_f=l_f/C=(l_d+l_h)/C$$

所以，连续ARQ的信道利用率为

$$U=\frac{D}{C}=\left(\frac{l_d}{l_d+l_h}\right)\left[\frac{1-p}{1+(\alpha-1)p}\right]\le 1$$

3.4 选择重传ARQ协议

选择重传ARQ的基本思想是，如果能只传送出现差错的或计时器超时的数据帧，就能进一步提高连续ARQ的信道利用率。

这就要求必须增大接收窗口，相应地也必须增大接收端的缓存空间，这是我们为了提高性能而需要付出的代价。

接收窗口的最大值

首先，接收窗口大于发送窗口没有意义，所以 $W_R\le W_T$。

其次，为了保证可靠传输，必须区分所有确认帧都丢失和都收到的极端情况，这就要求 $W_R+W_T\le 2^n$。

所以，接收窗口

$$W_R\le 2^n/2=2^{n-1} \quad (3\text{-}18)$$

3.5 面向比特的链路控制规程

3.5.1 HDLC协议概述

早期的数据链路控制规程如IBM的BSC规程（Binary Synchronous Communication）都是面向字符的。所谓**面向字符**就是说在链路上所传送的数据（包括数据部分和控制字符）必须是由规定字符集（如ASCII）中的字符组成。以BSC为例，其主要缺点是：

（1）开销比较大，使用停止等待协议，通信线路利用率低；

（2）所有设备必须使用同样的字符代码，不同版本的代码又不同；

（3）只对数据部分进行差错控制，可靠性较差；

（4）不灵活，不易扩展。每增加一种功能都要设定新的控制字符。

1974年，IBM推出的SNA的数据链路层采用了面向比特的规程SDLC（Synchronous Data Link Control）。后ISO把SDLC修改后，称为高级数据链路控制HDLC（High-level Data Link Control）。

3.5 面向比特的链路控制规程

三种站点

主站（primary station）控制整个链路的工作，它发出的帧叫做命令（command）。

次站（secondary station），又称从站，在主站的控制下工作，它发出的帧叫做响应（response）。

复合站（combined station），同时具有主站和次站的功能，可以发出命令和响应。

两种配置方式

非平衡配置，包括一个主站、一个或多个次站。分为点对点工作和多点工作两种情况。特点由主站控制整个链路的工作。

平衡配置，由两个复合站连接而成。

3.5 面向比特的链路控制规程

三种响应方式

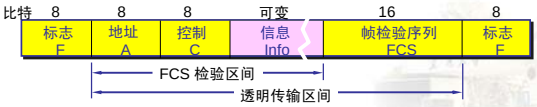
正常响应方式（Normal Response Mode, NRM），使用非平衡配置，只有主站才能发起数据传输，次站接受主站的探询（polling）。

异步响应方式（Asynchronous Response Mode, ARM），使用非平衡配置，允许次站向主站发起数据传输。很少使用。

异步平衡方式（Asynchronous Balanced Mode，ABM），使用平衡配置，每个站都可以平等的发起数据传输。

3.5.2 HDLC的帧结构

各字段的意义



3.5 面向比特的链路控制规程

物理层要解决比特同步问题，而数据链路层要解决帧同步问题。**标志字段F**（Flag）就是用来进行帧同步的。用**零比特填充法**实现透明传输。

数据中某一段比特组合恰好出现和 F 字段一样的情况

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

会被误认为是 F 字段

发送端在 5 个连 1 之后填入 0 比特再发送出去

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

填入 0 比特

在接收端将 5 个连 1 之后的 0 比特删除，恢复原样

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

在此位置删除填入的 0 比特

3.5 面向比特的链路控制规程

XI'AN JIAOTONG UNIVERSITY

地址字段A，使用非平衡方式时，总是写入次站地址；但在平衡方式时，总是填入确认站（应答站）的地址。全1是广播地址，全0是无效地址。地址字段可以扩展。

帧校验序列字段FCS，校验范围包括地址、控制、信息字段。

控制字段，是最复杂的字段。根据前两比特的值可以把HDLC帧分为**信息帧I**（Information）、**监督帧S**（Supervisory）和**无编号帧U**（Un-nummbered）三种。

0	N(S)		P/F	N(R)
1	0	S	P/F	N(R)
1	1	M	P/F	M

3.5 面向比特的链路控制规程

XI'AN JIAOTONG UNIVERSITY

信息帧

信息帧用来传输数据信息。N(S)表示当前发送的信息帧的编号，而N(R)表示所期望收到的帧的发送序号。由于N(R)字段的存在，我们就不必专门为收到的信息帧发送确认帧。在全双工通信时，可以让本站要发送的信息帧用N(R)捎带（piggybacking）确认序号。P/F在后面讨论。

监督帧

共有四种。为不能使用捎带技术的情况提供了ARQ机制。

RR（Receive Ready），确认N(R)-1及以前的帧，相当于ACK；

RNR（Receive Not Ready），暂停接收并确认N(R)-1及以前的帧；

REJ（Reject），否认从N(R)起的所有帧，确认N(R)-1及以前的帧；

SREJ（Selective Reject），只否认N(R)帧，确认N(R)-1及以前的帧。

3.5 面向比特的链路控制规程

XI'AN JIAOTONG UNIVERSITY

无编号帧

是指本身不带发送接收序号的帧。主要完成对链路的控制。

目前定义了15种，主要包括SNRM、SABM、SARM、DISC、UA、SNRME、SABME等用于连接管理的，另外还有FRMR（帧拒绝）等。

P/F比特

Poll/Final，查询/终止功能。为0无意义，为1时才有意义。

在非平衡配置正常响应下，次站只有收到主站发出的P比特为1的命令帧时才能发出响应。若有数据发送，则在最后一个数据帧把F置1；若无数据发送，则在应答的监督帧中把F置1。

在非平衡配置异步响应方式，或异步平衡方式中，任何一站均可在主动发送的S或I帧中将P置1，对方收到后，必须尽早回答本站状态，并将F置1，但并不表示数据已发完。

3.5 面向比特的链路控制规程

XI'AN JIAOTONG UNIVERSITY

基于平衡配置的异步平衡方式下的全双工通信举例。

图中所使用标注帧的格式为：

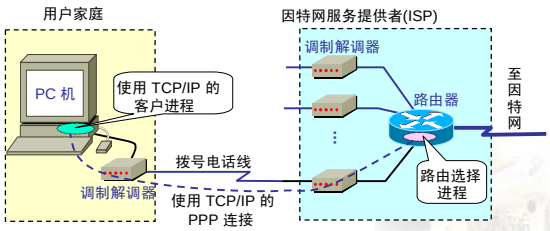
地址，帧名+N(S)+N(R)，P/F

3.6 因特网的点对点协议PPP

XI'AN JIAOTONG UNIVERSITY

3.6.1 PPP的工作原理

因特网的接入方式：拨号接入、专线接入。



3.6 因特网的点对点协议PPP

XI'AN JIAOTONG UNIVERSITY

早期大多使用SLIP（Serial Line Internet Protocol）协议，为了改进SLIP的缺点（如没有差错检测功能，不支持动态IP分配，不提供身份认证，未成为因特网的标准协议，存在多种互不兼容的版本），1992年制定了PPP协议。

PPP协议（RFC 1661）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，既支持异步链路（无奇偶校验的8比特），也支持面向比特的同步链路。

（2）一个用来建立、配置和测试数据连接的链路控制协议（Link Control Protocol, LCP ）。

（3）一套网络控制协议（Network Control Protocol, NCP），其中每个协议分别支持不同的网络层协议（如IP，DECnet，AppleTalk等）。

3.6 因特网的点对点协议PPP

XI'AN JIAOTONG UNIVERSITY

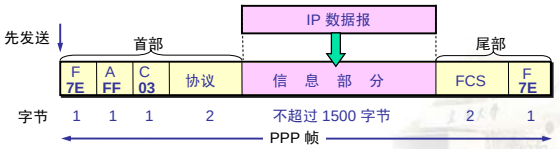
3.6.2 PPP的帧格式

如下图所示，PPP的帧格式与HDLC类似。

标志字段相同，为0x7E，即01111110B；

地址字段置为0xFF；控制字段只为0x03。

增加了协议字段，明信息字段的类型：0x0021表示IP数据报；0xC021表示链路控制数据； 0x8021表示网络控制数据。



3.6 因特网的点对点协议PPP

XI'AN JIAOTONG UNIVERSITY

PPP是面向字节的，所有PPP帧的长度都是整数个字节。

透明传输方法

当PPP用在同步传输链路时，规定用硬件完成零比特填充。

当PPP用在异步传输时，使用如下的特殊字符填充法：

（1）0x7E → 0x7D, 0x5E；

（2）0x7D → 0x7D, 0x5D；

（3）ASCII控制字符前加0x7D。

PPP不提供使用序号和确认的可靠传输。主要是基于如下考虑：

（1）在数据链路层出现差错的概率不大时，使用PPP较为合理。

（2）IP本身不可靠，链路层的可靠不能保证网络层的传输也可靠。

（3）帧校验序列FCS可以保证“无差错接受”。

3.6 因特网的点对点协议PPP

XI'AN JIAOTONG UNIVERSITY

3.6.3 PPP协议的工作状态

PPP的工作过程

（1）当用户拨号接入ISP时，通过modem和路由器建立一条物理连接。

（2）PC机向路由器发送一系列的LCP分组，这些分组及其响应选择了将要使用的一些PPP参数，即进行链路配置。

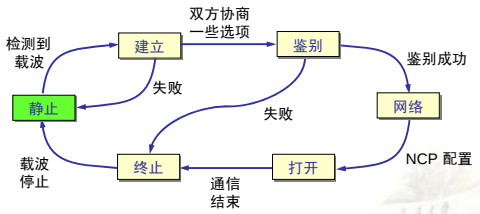
（3）通过NCP分组进行网络层配置，并给新接入的PC机分配一个临时的IP地址，使PC机成为因特网上的一个主机。

（4）通信完毕时，NCP 释放网络层连接，收回分配的IP地址。接着，LCP 释放数据链路层连接。最后modem释放物理层的连接。

3.6 因特网的点对点协议PPP

XI'AN JIAOTONG UNIVERSITY

PPP协议的状态图



XI'AN JIAOTONG UNIVERSITY