# 5.5 协议说明与验证（1）

## 5.5.1 通信协议中的形式化描述技术

- 形式化描述的意义

  实际使用的协议非常复杂，给协议的理解、验证、实现和测试等工作带来困难，需要采用形式化的、数学的描述方法来描述协议。但是目前大多数协议还是采用自然语言描述。

- 自然语言描述协议的缺点
  - 冗余；
  - 多义性；
  - 结构性不好；
  - 不便于自动验证、测试、实现。

# 5.5 协议说明与验证（2）

- 形式化描述技术FDT（Formal Description Technique）/形式化方法FM（Formal Method）广泛应用于协议工程研究中
  - 协议说明（Protocol Specification）
  - 协议验证（Protocol Verification）
  - 协议实现（Protocol Implementation）
  - 协议测试（Protocol Testing）
    - 一致性测试（Conformance Testing）
    - 互操作性测试（Cooperability Testing）
    - 性能测试（Performance Testing）
- 协议说明

  必须既定义一个协议实体提供给它的用户的服务，又定义该协议实体的内部操作。

# 5.5 协议说明与验证（3）

- 协议验证

  验证协议说明是否完整、正确。
- 协议实现

  用硬件和/或软件实现协议说明中规定的功能。
- 协议测试

  用测试的方法来检查协议实现是否满足要求，包括：协议实现是否与协议说明一致（一致性测试）、协议实现之间的互操作能力（互操作性测试）和协议实现的性能（性能测试）等。
- 在协议的说明、验证、实现和测试过程中使用形式化描述技术，不仅可以比较容易地理解协议，而且可以使协议描述更加精确，大大简化了协议的研究工作。

# 5.5 协议说明与验证（4）

- 一种形式化方法总是以一种形式体系为基础，只是在具体应用时，大都做了便于描述的改进和扩充。
- 常用的形式化方法
  - 有限状态机FSM（Finite State Machine）
    扩展：EFSM
  - 形式化语言模型
    LOTOS，Estelle，SDL
    都有相应扩展
  - Petri网
    扩展：时间Petri网，随机Petri网，高级Petri网
  - 过程代数（Process Algebra）
    扩展：随机过程代数
- 形式化方法的主要问题是状态爆炸

# 5.5 协议说明与验证（5）

## 5.5.2 有限状态机模型

- 定义

  一个有限状态机是一个四元组(S, M, I, T)，其中

  S是状态的集合；

  M是标号的集合；

  I是初始状态的集合；

  T是变迁的集合。

- 通信协议建模
  - 基本出发点：认为通信协议主要是由响应多个“事件”的相对简单的处理过程组成；
  - 事件
    - 命令（来自用户）
    - 信息到达（来自低层）
    - 内部超时

# 5.5 协议说明与验证（6）

- 优点：简单明了，比较精确；
- 缺点：对许多复杂的协议，事件数和状态数会剧增，处理困难。

– 例 协议3

每个状态用三个字母表示：XYZ

X：发送方正发送的帧序号，为0或1；

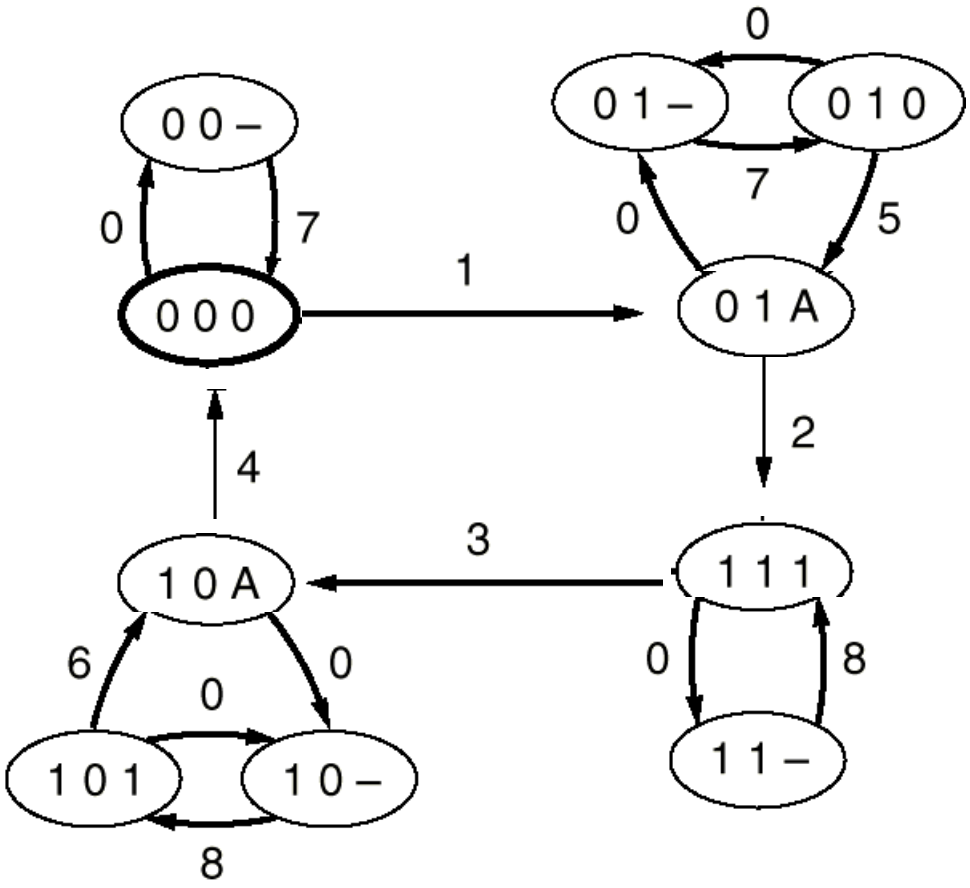
Y：接收方正等待的帧序号，为0或1；

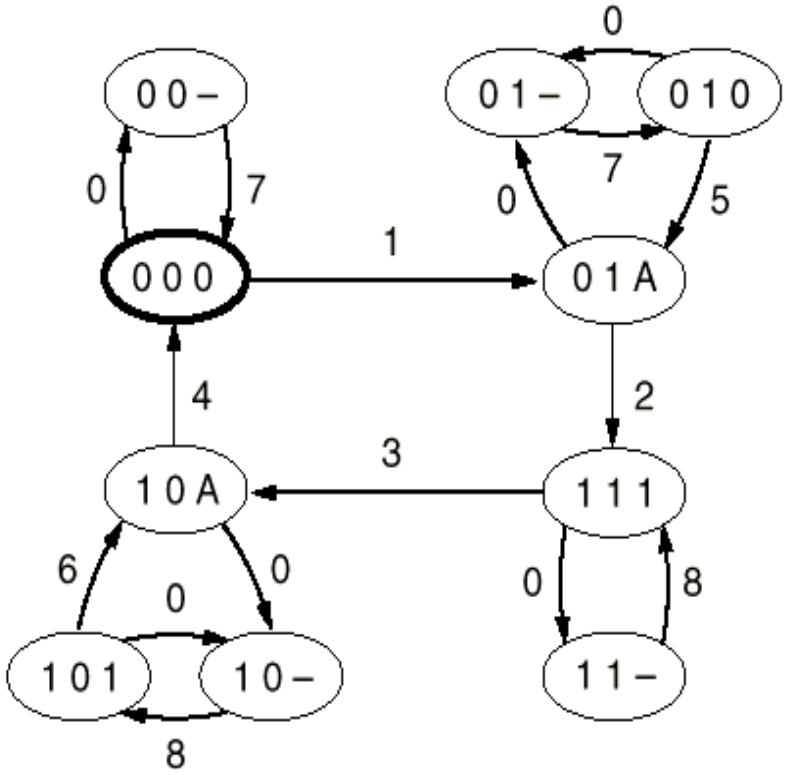
Z：信道状态，为0，1，A或 -（空）。

初始状态为（000）

半双工信道　　Fig. 3-20

全双工信道　　Fig. 3-21


0 0 –
0 1 –
0 1 0
0 0 0
0 1 A
1 1 1
1 0 A
1 0 1
1 0 –
1 1 –
0
7
0
5
0
7
1
2
3
4
6
0
0
8
0
8

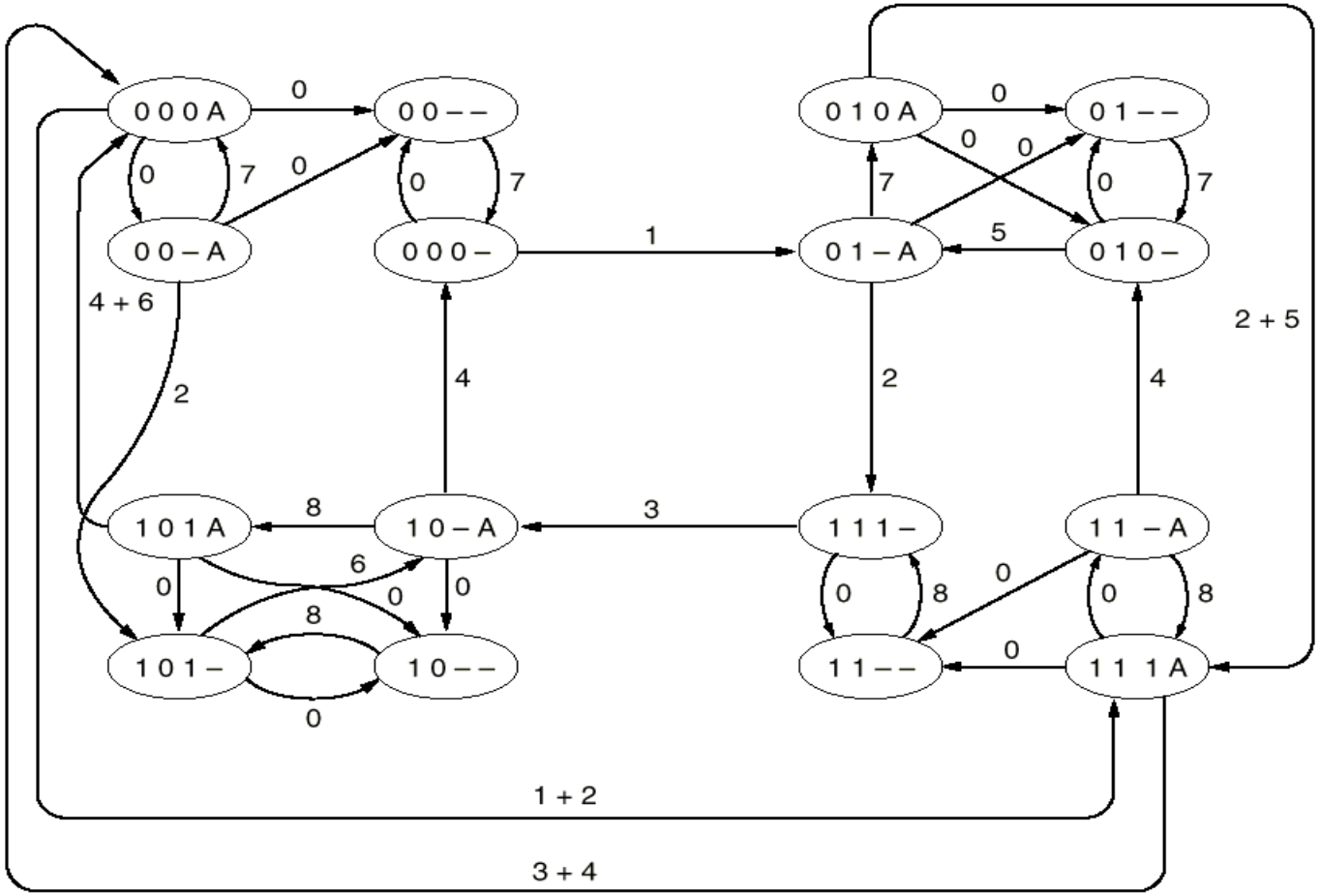



(a)

| Transition | Who runs? | Frame accepted | Frame emitted | To network layer |
|---|---|---|---|---|
| 0 | – | (frame lost) | | – |
| 1 | R | 0 | A | Yes |
| 2 | S | A | 1 | – |
| 3 | R | 1 | A | Yes |
| 4 | S | A | 0 | – |
| 5 | R | 0 | A | No |
| 6 | R | 1 | A | No |
| 7 | S | (timeout) | 0 | – |
| 8 | S | (timeout) | 1 | – |

(b)

Fig. 3-20. (a) State diagram for protocol 3. (b) Transitions.




(a)

(0 0 0 –), (0 1 – A), (0 1 0 A), (1 1 1 A), (1 1 – A), (0 1 0 –), (0 1 – A), (1 1 1 –)

(b)

Fig. 3-21. (a) State graph for protocol 3 and a full-duplex channel. (b) Sequence of states causing the protocol to fail.

# 5.5 协议说明与验证（7）

- 协议验证
  - 验证协议说明是否完整正确，以协议说明为基础，涉及逻辑证明。
  - 主要用于系统实现前的设计阶段，为了避免可能出现的设计错误。原则上验证涉及协议所有可能的状态。
  - 可达性分析是一种常用的验证方法

    利用图论知识可以解决状态的可达性问题；

    可达性分析能够用来解决协议的不完整性、死锁和无关变迁等问题。

# 5.5 协议说明与验证（8）

## 5.5.3 Petri网模型

Petri网模型最早在1962年 Carl Adam Petri的博士论文中提出来，主要特性是具有较强的对并行、不确定性、异步和分布的描述能力和分析能力。

– Petri网研究的系统模型行为特性包括
  - 状态的可达（reachability）
  - 位置的限界（boundedness）
  - 变迁的活性（liveness）
  - 初始状态的可逆达（reversibility）
  - 标识（marking）之间的可达（reachability）
  - 变迁之间的坚挺（persistence）
  - 事件之间的同步距离（synchronic distance）
  - 公平性（fairness）

# 5.5 协议说明与验证（9）

- 定义
  - 一个Petri网的结构元素包括：
    - 位置（place）：描述系统状态，用一个圆圈表示；
    - 变迁（transition）：描述修改系统状态的事件，用一个长方形或线段表示；
    - 弧（arc）：描述状态与事件之间的关系，包括输入弧和输出弧，用有向弧表示。
  - 活动元素 —— 标记（token）：
    - 包含在位置中，用点表示；
    - 用来表示处理的信息单元、资源单元和顾客、用户等对象；
    - 如果位置用来描述条件，它可以包含一个标记或不包含标记，当包含标记时，条件为真，否则，为假；
    - 如果位置用来定义状态，位置中的标记个数用于规定这个状态；

# 5.5 协议说明与验证（10）

- 变迁实施规则（firing rule）：
  - 如果一个变迁的所有输入位置至少包含一个标记，则这个变迁可能实施；
  - 一个可实施变迁的实施导致从它所有输入位置中都清除一个标记，在它所有输出位置中产生一个标记；
  - 当使用大于1的弧权（weight）时，在变迁的所有输入位置中都要包含至少等于连接弧权的标记个数它才可实施，并根据弧权，在每个输出位置中产生相应标记个数；
  - 变迁的实施是一个原子操作，输入位置清除标记和输出位置产生标记是一个不可分割的完整操作。

Fig. 3-22

– 主要分析方法
  - 可达树
  - 关联矩阵和状态方程
  - 不变量
  - 分析化简规则

# 5.5 协议说明与验证（11）

- Petri网的扩展
  - 条件/事件（C/E）网
    最简单，每个位置最多一个标记，表示条件。
  - 位置/变迁（P/T）网
    每个位置中的标记可以有多个。
  - 高级Petri网（包括谓词/变迁网和着色网）
    给标记以属性，即标记有区别
  - 从没有参数的网，发展到时间Petri网和随机Petri网；
  - 从一般有向弧发展到可变弧；
  - 从自然数标记个数发展到概率标记个数；
  - 从原子变迁发展到谓词变迁和子网变迁。

Petri网这个上课可能提到的原理还需要再仔细看一下。

# 5.5 协议说明与验证（12）

– 例：协议3（半双工信道）

Fig. 3-23

可达图（部分）

– 关于形式化方法的几点注意事项

- 模型描述能力的增强会在某种程度上增加模型分析的难度；
- 模型仅仅是手段，不是目的。

  Modeling is a bridge.

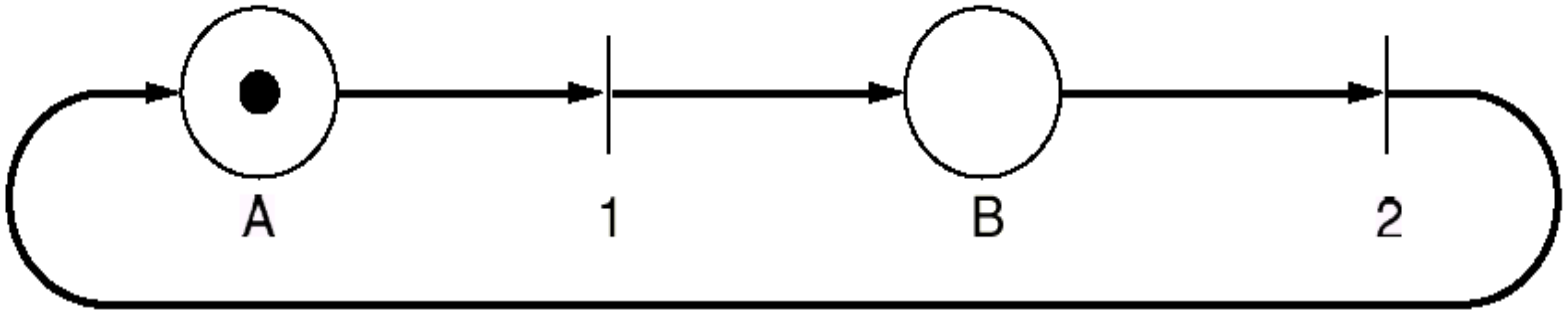



Fig. 3-22. A Petri net with two places and two transitions.

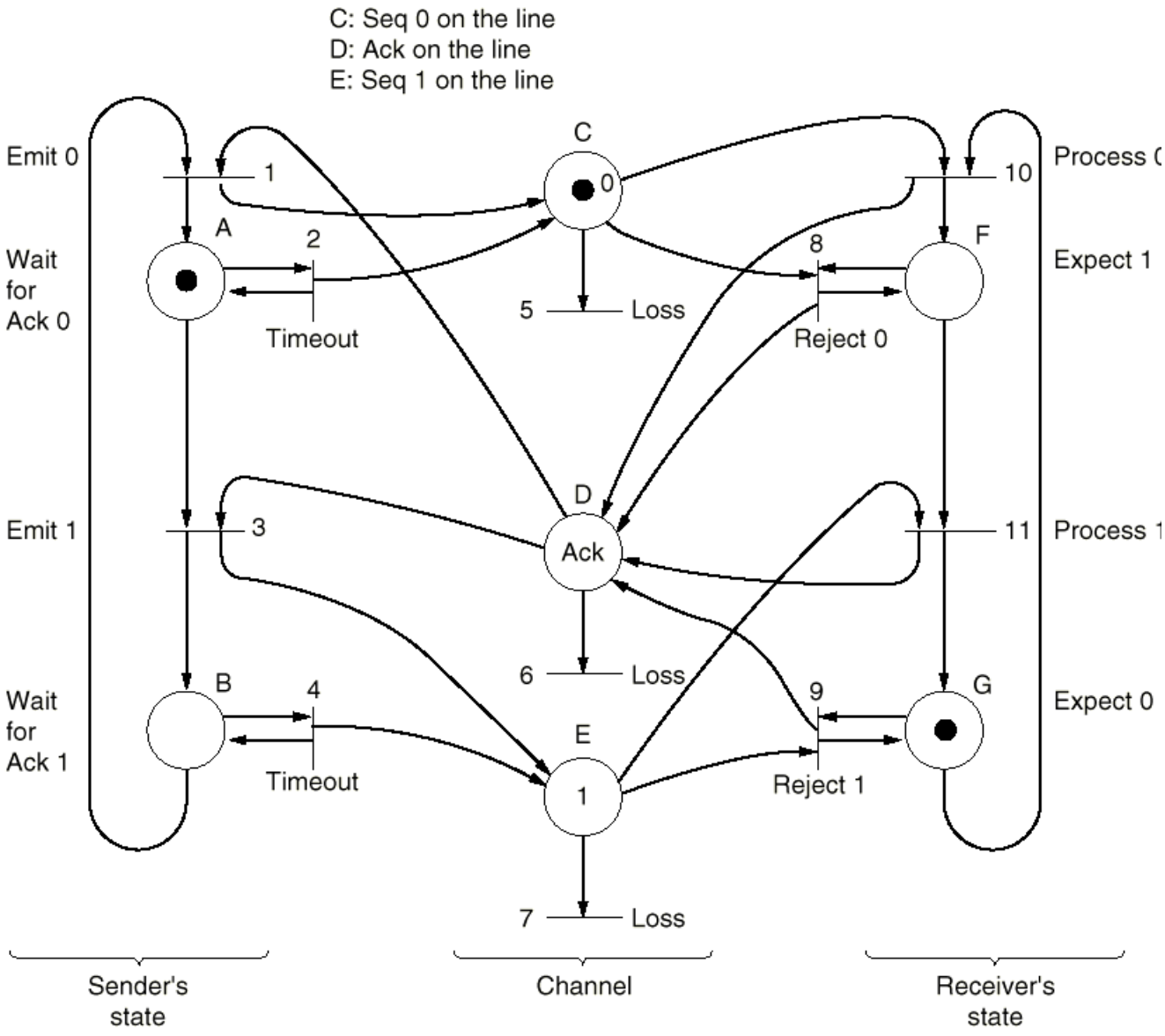
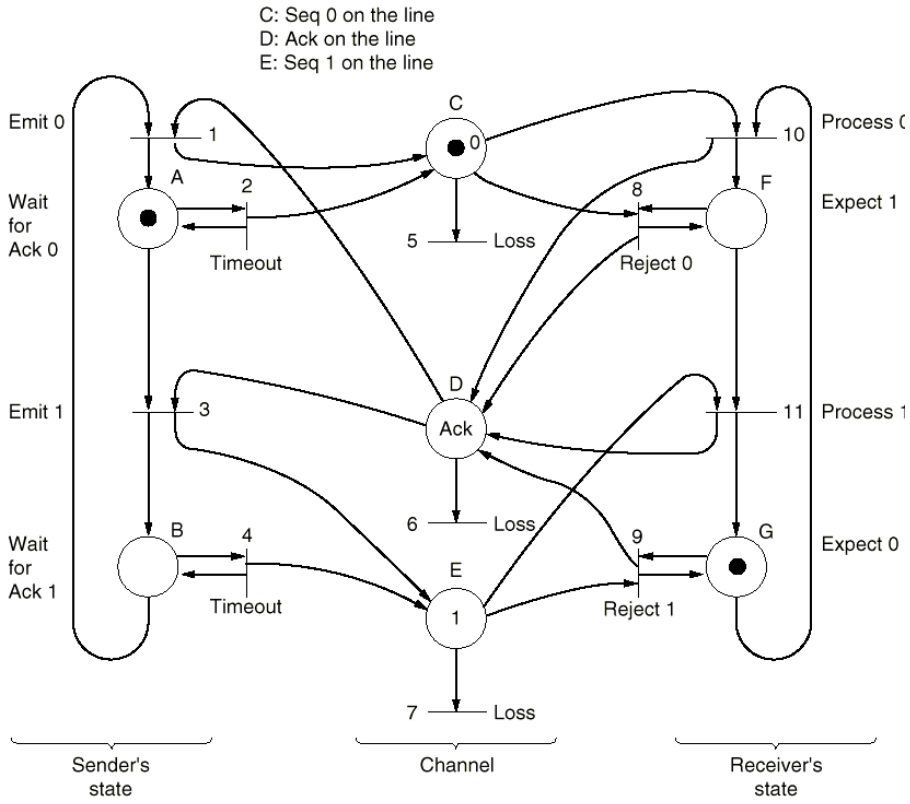



Fig. 3-23. A Petri net model for protocol 3.



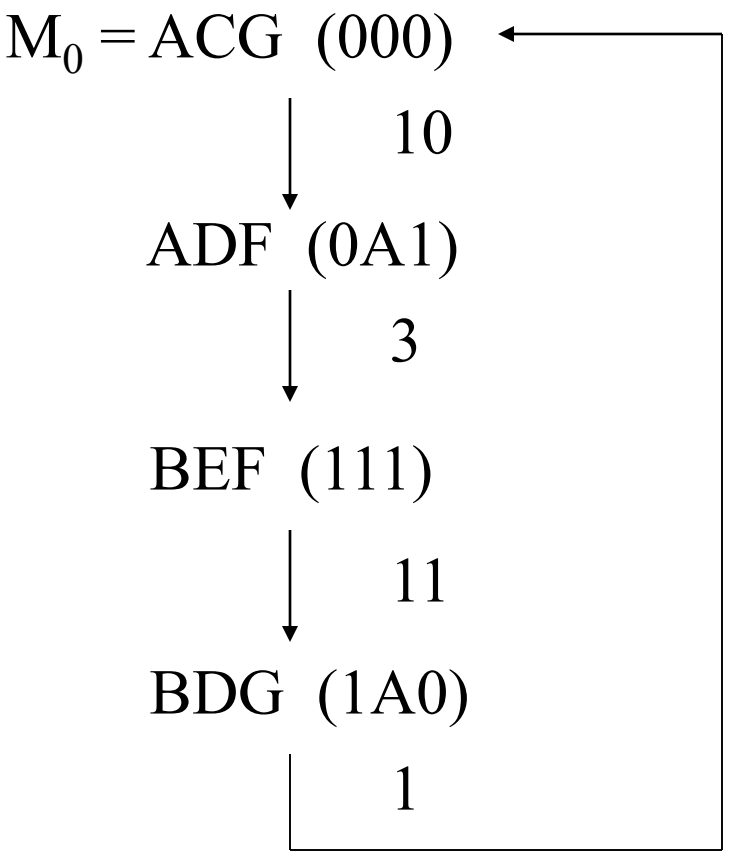




Fig. 3-23. A Petri net model for protocol 3.

# 5.6 常用的数据链路层协议（1）

ISO和CCITT在数据链路层协议的标准制定方面做了大量工作，各大公司也形成了自己的标准。

- 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）
  - 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。

# 5.6 常用的数据链路层协议（2）

- 数据链路层协议比较

| | 面向字符的链路层协议 | 面向比特的链路层协议 |
|---|---|---|
| 报文格式 | 信息报文和监控报文的格式不统一，控制复杂。 | 采用统一的帧格式，信息报文和监控报文均以帧为单位传输，控制简单统一。 |
| 透明性 | 报文中不允许出现控制字符，透明性差。字符填充。 | 编码独立，传输透明。不受任何比特式样和字符宽度的限制。比特填充 |
| 可靠性 | 只做奇偶校验，可靠性差。 | CRC 校验，可靠性高。 |
| 发送方式 | 等待发送 | 连续发送 |
| 纠错方式 | 停-等重发 | 退后 n 帧重发或选择重发 |
| 传输效率 | 低 | 高 |
| 灵活性 | 对每种应用形式不同 | 同一种形式适用于所有场合 |

# 5.6 常用的数据链路层协议（3）

## 5.6.1 高级数据链路控制规程HDLC

1976年，ISO提出HDLC（High-level Data Link Control）

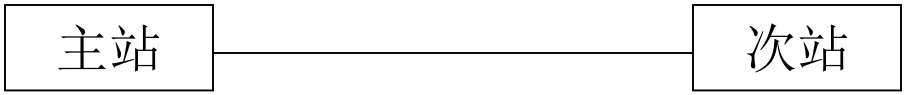
- HDLC的组成
  - 帧结构
  - 规程元素 } 语法
  - 规程类型 语义

  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

# 5.6　常用的数据链路层协议（4）

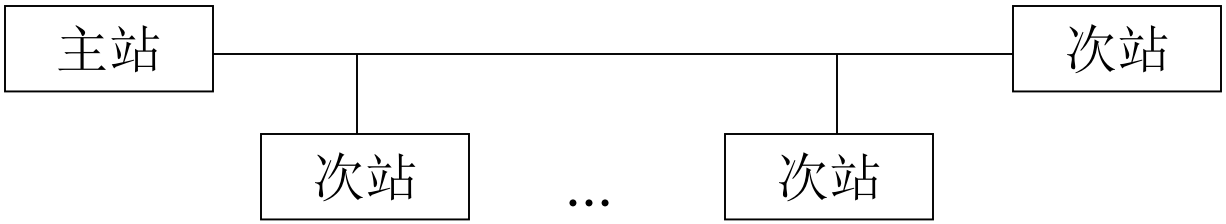
- 数据站（简称站 station），由计算机和终端组成，负责发送和接收帧。HDLC涉及三种类型的站：
  - 主站（primary station）：主要功能是发送命令（包括数据），接收响应，负责整个链路的控制（如系统的初始、流控、差错恢复等）；
  - 次站（secondary station）：主要功能是接收命令，发送响应，配合主站完成链路的控制；
  - 组合站（combined station）：同时具有主、次站功能，既发送又接收命令和响应，并负责整个链路的控制。
- HDLC适用的链路构型
  - 非平衡型
    - 点 — 点式

HDLC协议的细节需要考虑一下。



# 5.6　常用的数据链路层协议（5）

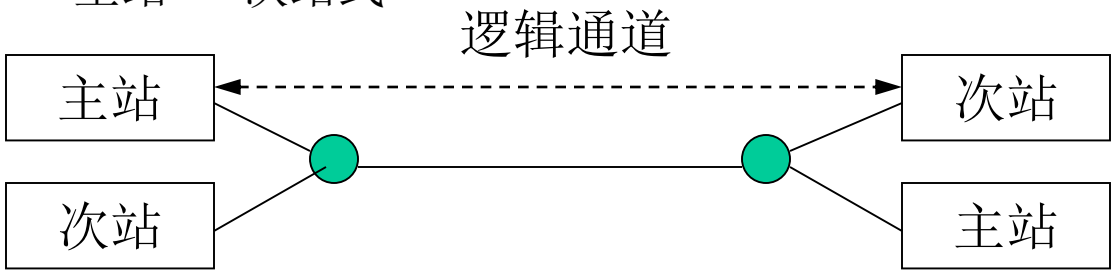
- 多点式



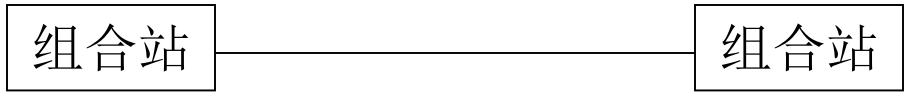

适合把智能和半智能的终端连接到计算机。

– 平衡型

- 主站 — 次站式




- 组合式




适合于计算机和计算机之间的连接

# 5.6 常用的数据链路层协议（6）

- HDLC的基本操作模式
  - 正规响应模式 NRM（Normal Response Mode）

    适用于点 — 点式和多点式两种非平衡构型。只有当主站向次站发出探询后，次站才能获得传输帧的许可。
  - 异步响应模式 ARM（Asynchronous Response Mode）

    适用于点 — 点式非平衡构型和主站 — 次站式平衡构型。次站可以随时传输帧，不必等待主站的探询。
  - 异步平衡模式 ABM（Asynchronous Balanced Mode）

    适用于通信双方都是组合站的平衡构型，也采用异步响应，双方具有同等能力。
- 帧结构

  Fig. 3-24
  - 地址域（Address）
    - 多终端线路，用来区分终端；

# 5.6 常用的数据链路层协议（7）

- 点到点线路，有时用来区分命令和响应。
  若A是接收该帧的站的地址，则该帧是命令帧；
  若A是发送该帧的站的地址，则该帧是响应帧。

– 控制域（Control）
  - 序号
    使用滑动窗口技术，3位序号，发送窗口大小为7
  - 确认
  - 其它

– 数据域（Data）
  - 任意信息，任意长度（有上限）

– 校验和（Checksum）
  - CRC校验
  - 生成多项式：CRC-CCITT

– 定界符
  - 01111110
  - 空闲的点到点线路上连续传定界符

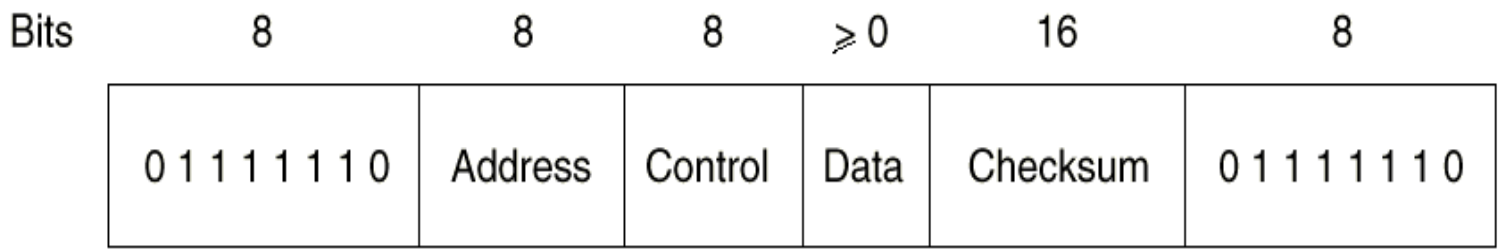



Fig. 3-24. Frame format for bit-oriented protocols.

# 5.6 常用的数据链路层协议（8）

- 帧类型
  - 信息帧（Information）
  - 监控帧（Supervisory）
  - 无序号帧（Unnumbered）
- 控制域

  Fig. 3-25
  - 序号（Seq）
    - 使用滑动窗口技术，3位序号，发送窗口大小为7
  - 捎带确认（Next）
    - 捎带第一个未收到的帧序号，而不是最后一个已收到的帧序号
  - 探询/结束 P/F位（Poll/Final）
    - 命令帧置P位，响应帧置F位。有些协议，P/F位用来强迫对方机器立刻发控制帧；
    - 多终端，终端发向计算机的帧中，最后一个帧P/F位置为“F”，其它置为“P”。

# 5.6 常用的数据链路层协议（9）

- 类型（Type）
  - “0”表示确认帧 RR（RECEIVE READY）；
  - “1”表示否定性确认帧 REJ（REJECT）。
  - “2”表示接收未准备好 RNR（RECEIVE NOT READY）
  - “3”表示选择拒绝 SREJ（SELECTIVE REJECT）

    HDLC和ADCCP允许选择拒绝，SDLC和LAPB不允许。

- 无序号帧

  可以用来传控制信息，也可在不可靠无连接服务中传数据。

- 命令
  - DISC（DISConnect）

    拆除连接请求
  - SNRM（Set Normal Response Mode）
  - SARM（Set Asynchronous Response Mode）

# 5.6 常用的数据链路层协议（10）

- SABM（Set Asynchronous Balanced Mode）
  HDLC和LAPB使用。
- SABME
- SNRME
- FRMR（FRaMe Reject）
  校验和正确，语义错误
- 无序号确认UA（Unnumbered Acknowledgement）
  对控制帧进行确认，用于确认模式建立和接受拆除命令。
- UI（Unnumbered Information）

- HDLC的功能组合
  - 三种站，两种构型，三种操作模式，以及规程元素中定义的各种帧的各种组合产生多种链路层协议。
  - HDLC定义了选择构成链路层协议的良序结构：
    选择站构型 ——> 基本操作模式 ——> 基本帧种类 ——> 12种任选功能 ——> 得到协议

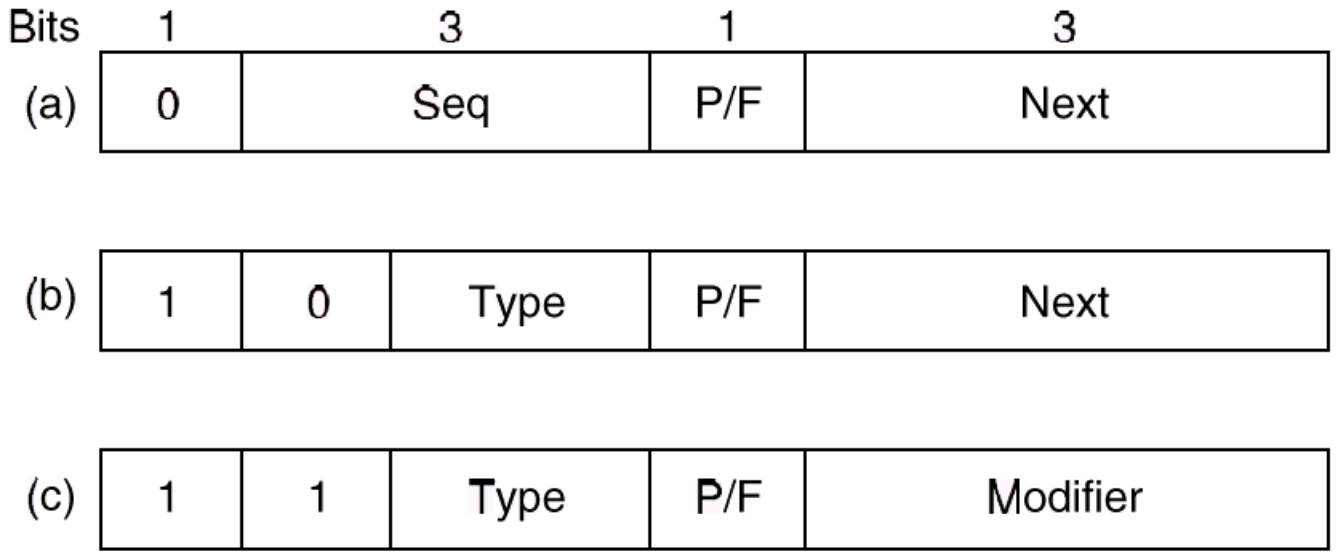



Fig. 3-25. Control field of (a) an information frame, (b) a supervisory frame, (c) an unnumbered frame.

# 5.6　常用的数据链路层协议（11）

## 5.6.2 X.25的链路层协议LAPB

- X.25协议
  - 分组级，PLP
  - 帧级，X.25 LAP（Link Access Procedure），X.25 LAPB（Balanced）
  - 物理级，X.21
- “X.25协议规程使用HDLC规程的原理和术语”
- X.25 LAP：HDLC非平衡规程帧的基本清单 + 任选功能2、8、12，也可组成主站 — 次站式平衡规程。
- X.25 LAPB：HDLC组合站平衡规程帧的基本清单 + 任选功能2、8、11、12。
- 因此，X.25 LAP、LAPB是HDLC的子集。

# 5.6 常用的数据链路层协议（12）

- X.25的帧格式与HDLC完全相同
- X.25链路级的命令和响应

| 格式 | 命令 | 响应 | 控制域编码 | | | |
|---|---|---|---|---|---|---|
| 信息帧 | I(信息) | | 0 | N(S) | P | N(R) |
| 监控帧 | RR(接收准备好)<br>RNR(接收未准备好)<br>REJ(拒绝) | RR(接收准备好)<br>RNR(接收未准备好)<br>REJ(拒绝) | 1 000<br>1 010<br>1 011 | | P/F<br>P/F<br>P/F | N(R)<br>N(R)<br>N(R) |
| 无序号帧 | SARM(置异步响应模式) | DM(拆除模式) | 1 111 | | P/F | 000 |
| | SABM(置异步平衡模式) | | 1 111 | | P | 100 |
| | DISC(拆除) | | 1 100 | | P | 010 |
| | | UA(无序号确认) | 1 100 | | F | 110 |
| | | CMDR(命令拒绝)<br>FRMR(帧拒绝) | 1 110 | | F | 001 |

# 5.6　常用的数据链路层协议（13）

- X.25 LAPB的各种检错和纠错措施

a 帧格式上采用CRC校验，只检错，不纠错，丢弃出错帧；

b 设立超时机制，计时器

超时重传，重传N次，则向上层协议报告。

超时机制用来检错，重传用来纠错。

c 帧序号

若接收方发现帧序号错，就发拒绝帧给发送方，发送方重传，既检错也纠错。

d 采用P/F位来进行校验指示

发送置为 P 的命令帧，等待置为 F 的响应帧，能及时发现远程数据站是否收到命令帧。

规程规定：a 必须使用；b, c, d 组合使用。

# 5.6 常用的数据链路层协议（14）

## 5.6.3 Internet数据链路层协议

- 点到点通信的两种主要情形
  - 路由器到路由器（router-router leased line connection）
  - 通过modem拨号上网，连到路由器或接入服务器（Access Server）（dial-up host-router connection）

    Fig. 3-26
- SLIP —— Serial Line IP
  - 1984年，Rick Adams提出，RFC1055，发送原始IP包，用一个标记字节来定界，采用字符填充技术；
  - 新版本提供TCP和IP头压缩技术，RFC 1144
  - 存在的问题
    - 不提供差错校验
    - 只支持IP
    - IP地址不能动态分配
    - 不提供认证
    - 多种版本并存，互连困难

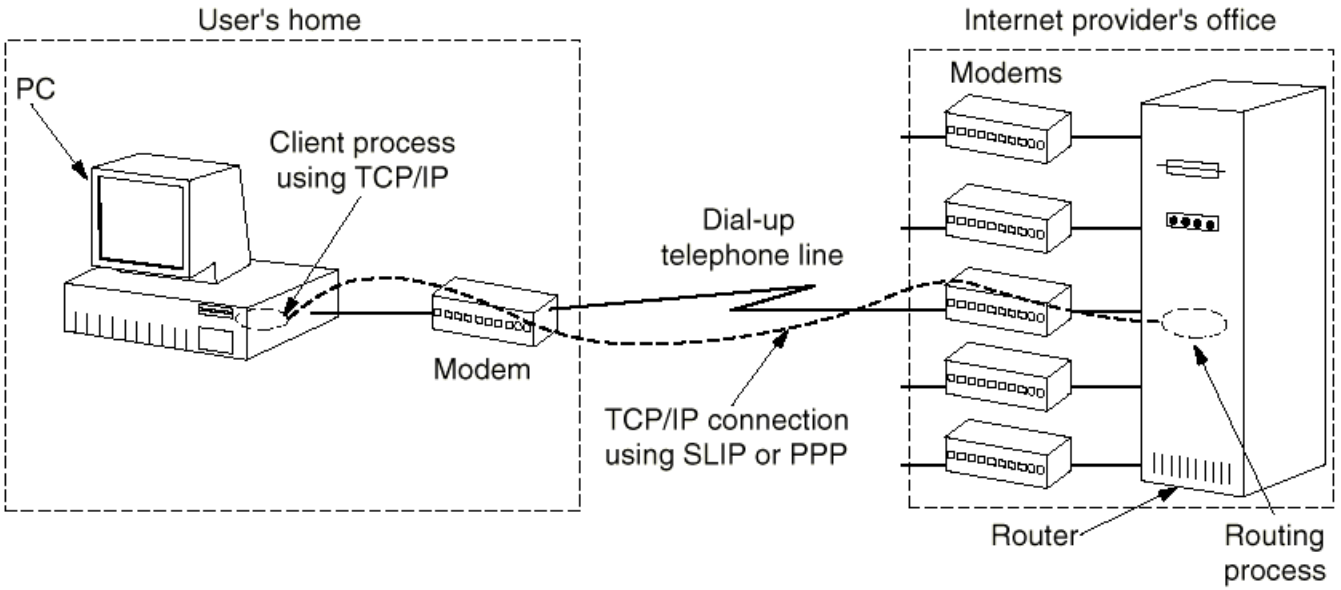



Fig. 3-26. A home personal computer acting as an Internet host.

# 5.6 常用的数据链路层协议（15）

- 点到点协议 PPP —— Point-to-Point Protocol
  - RFC 1661，RFC 1662，RFC 1663
  - 与SLIP相比，PPP有很大的提高，提供差错校验、支持多种协议、允许动态分配IP地址、支持认证等。
  - 以帧为单位发送，而不是原始IP包；
  - 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

    Fig. 3-27

# 5.6 常用的数据链路层协议（16）

- 标记域：01111110，字符填充；
- 地址域：11111111
- 控制域：缺省值为00000011，表示无序号帧，不提供使用序号和确认的可靠传输；不可靠线路上，也可使用有序号的可靠传输。
- 协议域：指示净负荷中是何种包
- 净负荷域：变长，缺省为1500字节；
- 校验和域：2或4个字节

– LCP帧类型

Fig. 3-29

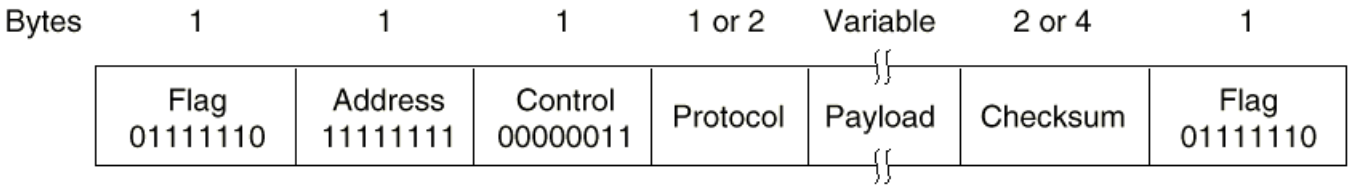



Fig. 3-27. The PPP full frame format for unnumbered mode operation.

| Name | Direction | Description |
|---|---|---|
| Configure-request | I → R | List of proposed options and values |
| Configure-ack | I ← R | All options are accepted |
| Configure-nak | I ← R | Some options are not accepted |
| Configure-reject | I ← R | Some options are not negotiable |
| Terminate-request | I → R | Request to shut the line down |
| Terminate-ack | I ← R | OK, line shut down |
| Code-reject | I ← R | Unknown request received |
| Protocol-reject | I ← R | Unknown protocol requested |
| Echo-request | I → R | Please send this frame back |
| Echo-reply | I ← R | Here is the frame back |
| Discard-request | I → R | Just discard this frame (for testing) |

Fig. 3-29. The LCP packet types.