用广播对所有设备提问：“××这个IP地址是谁的？请把你的MAC地址告诉我。”然后就会有人回答：“这个IP地址是我的，我的MAC地址是×××××。”[①]（图2.19）

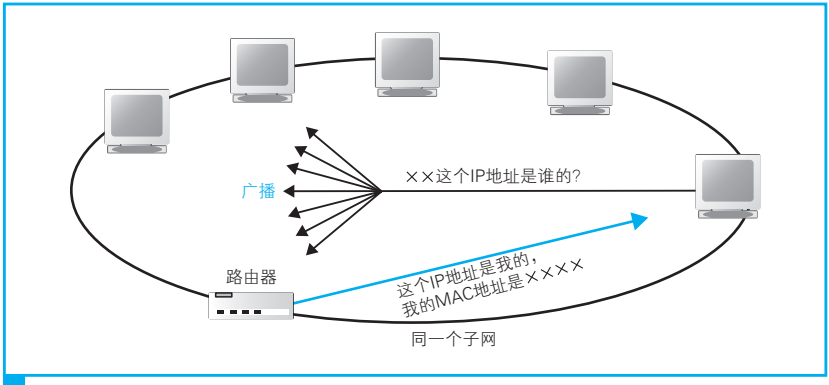


图2.19 用ARP查询MAC地址

如果对方和自己处于同一个子网中，那么通过上面的操作就可以得到对方的MAC地址[②]。然后，我们将这个MAC地址写入MAC头部，MAC头部就完成了。

不过，如果每次发送包都要这样查询一次，网络中就会增加很多ARP包，因此我们会将查询结果放到一块叫作ARP缓存的内存空间中留着以后用。也就是说，在发送包时，先查询一下ARP缓存，如果其中已经保存了对方的MAC地址，就不需要发送ARP查询，直接使用ARP缓存中的地址，而当ARP缓存中不存在对方MAC地址时，则发送ARP查询。显示ARP缓存的方法和MAC地址的写法如图2.20和图2.21所示，供大家参考。

① 不是这个IP地址的设备会忽略广播，什么都不回答。

② 如果路由表的设置正确，那么对方应该在同一子网，否则对方无法作出ARP响应，这时只能认为对方不存在，包的发送操作就会失败。

显示ARP缓存内容的命令。像ARP –d 10.10.1.43这样加上–d选项表示删除ARP缓存中保存的条目

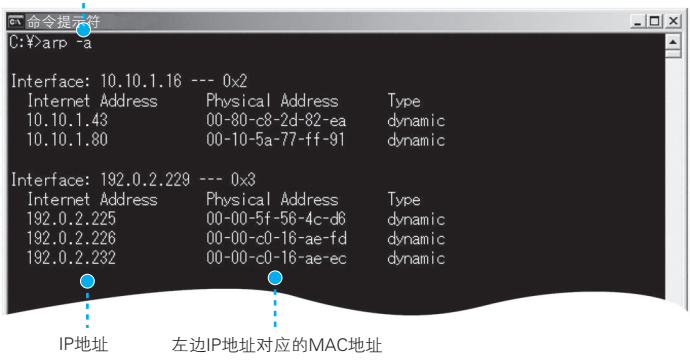


图 2.20　ARP 缓存的内容

（a）用“-”分隔的写法

00-80-C8-2D-82-EA

（b）用“:”分隔的写法

00:80:C8:2D:82:EA

MAC地址长度为48比特（6字节），按照惯例有（a）、（b）两种写法，它们的意思是一样的，使用任何一种写法都可以。

图 2.21　MAC 地址

有了 ARP 缓存，我们可以减少 ARP 包的数量，但如果总是使用 ARP 缓存中保存的地址也会产生问题。例如当 IP 地址发生变化时，ARP 缓存的内容就会和现实发生差异。为了防止这种问题的发生，ARP 缓存中的值在经过一段时间后会被删除，一般这个时间在几分钟左右。这个删除的操作非常简单粗暴，不管 ARP 缓存中的内容是否有效，只要经过几分钟就全部删掉，这样就不会出问题了。当地址从 ARP 缓存中删除后，只要重新执

行一次 ARP 查询就可以再次获得地址了。

上面这个策略能够在几分钟后消除缓存和现实的差异，但 IP 地址刚刚发生改变的时候，ARP 缓存中依然会保留老的地址，这时就会发生通信的异常[①]。

查询 MAC 地址需要使用 ARP。

将 MAC 头部加在 IP 头部的前面，整个包就完成了。到这里为止，整个打包的工作是由 IP 模块负责的。有人认为，MAC 头部是以太网需要的内容，并不属于 IP 的职责范围，但从现实来看，让 IP 负责整个打包工作是有利的。如果在交给网卡之前，IP 模块能够完成整个打包工作，那么网卡只要将打好的包发送出去就可以了。对于除 IP 以外的其他类型的包也是一样，如果在交给网卡之前完成打包，那么对于网卡来说，发送的操作和发送 IP 包是完全相同的。这样一来，同一块网卡就可以支持各种类型的包。至于接收操作，我们到后面会讲，但如果接收的包可以原封不动直接交给 IP 模块来处理，网卡就只要负责接收就可以了。这样一来，一块网卡也就能支持各种类型的包了。与其机械地设计模块和设备之间的分工，导致网卡只能支持 IP 包，不如将分工设计得现实一些，让网卡能够灵活支持各种类型的包。

2.5.6 以太网的基本知识

完成 IP 模块的工作之后，下面就该轮到网卡了，不过在此之前，我们先来了解一些以太网的基本知识。

以太网是一种为多台计算机能够彼此自由和廉价地相互通信而设计的通信技术，它的原型如图 2.22（a）所示。从图上不难看出，这种网络的本质其实就是一根网线。图上还有一种叫作收发器的小设备，它的功能只是将不同网线之间的信号连接起来而已。因此，当一台计算机发送信号时，信号就会通过网线流过整个网络，最终到达所有的设备。这就好像所有人

① 遇到这种情况，可以查看 ARP 缓存的内容，并手动删除过时的条目。

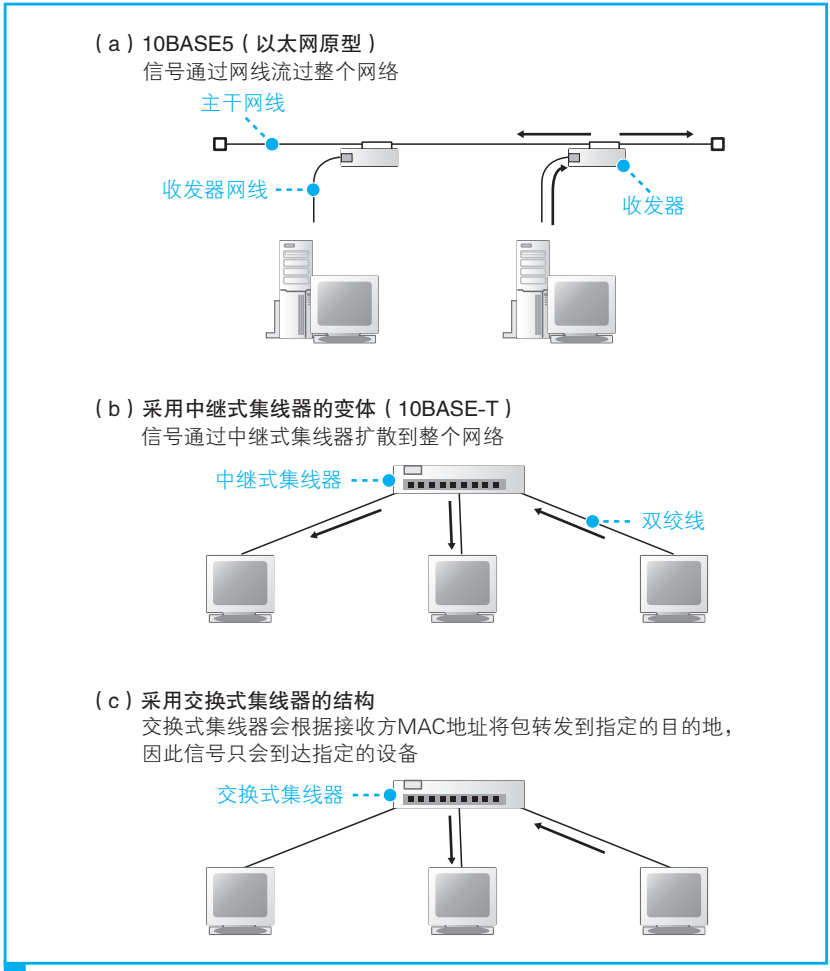


图 2.22　以太网的基本结构

待在一个大房间里，任何一个人说话，所有人都能够听到，同样地，这种网络中任何一台设备发送的信号所有设备都能接收到。不过，我们无法判断一个信号到底是发给谁的，因此需要在信号的开头加上接收者的信息，也就是地址。这样一来就能够判断信号的接收者了，与接收者地址匹配的设备就接收这个包，其他的设备则丢弃这个包，这样我们的包就送到指定

的目的地了。为了控制这一操作，我们就需要使用表 2.3 中列出的 MAC 头部。通过 MAC 头部中的接收方 MAC 地址，就能够知道包是发给谁的；而通过发送方 MAC 地址，就能够知道包是谁发出的；此外，通过以太类型就可以判断包里面装了什么类型的内容。以太网其实就这么简单[①]。

这个原型后来变成了图 2.22（b）中的结构。这个结构是将主干网线替换成了一个中继式集线器[②]，将收发器网线替换成了双绞线[③]。不过，虽然网络的结构有所变化，但信号会发送给所有设备这一基本性质并没有改变。

后来，图 2.22（c）这样的使用交换式集线器[④]的结构普及开来，现在我们说的以太网指的都是这样的结构。这个结构看上去和（b）很像，但其实里面有一个重要的变化，即信号会发送给所有设备这一性质变了，现在信号只会流到根据 MAC 地址指定的设备，而不会到达其他设备了。当然，根据 MAC 地址来传输包这一点并没有变，因此 MAC 头部的设计也得以保留。

尽管以太网经历了数次变迁，但其基本的 3 个性质至今仍未改变，即将包发送到 MAC 头部的接收方 MAC 地址代表的目的地，用发送方 MAC 地址识别发送方，用以太类型识别包的内容。因此，大家可以认为具备这 3 个性质的网络就是以太网[⑤]。

---

① 实际上，多台设备同时发送信号会造成碰撞，当然也有相应的解决方案，不过这部分比较复杂。随着交换式集线器的普及，信号已经不会发生碰撞了，因此在实际工作中也不需要在意这个复杂的部分。

② 中继式集线器：在以太网（10BASE-T/100BASE-TX）中简称集线器。如果需要区分仅对信号进行放大中继的传统集线器和交换式集线器，则将前者称为中继式集线器，也叫共享式集线器。我们将在 3.1.4 一节进行介绍。（以下将“中继式集线器”简称为“集线器”——译者注）

③ （a）和（b）中流过的信号不同，因此单纯的替换似乎有点简单粗暴。

④ 以下将“交换式集线器”简称为“交换机”。——译者注

⑤ 这些性质也适用于无线局域网。也就是说，将包发送到 MAC 头部的接收方 MAC 地址所代表的目的地，用发送方 MAC 地址识别发送方，在这些方面无线局域网和以太网是一样的。无线局域网没有以太类型，但有另一个具备同样功能的参数，可以认为它就是以太类型。因此，我们可以用无线局域网来代替以太网。

以太网中的各种设备也是基于以太网规格来工作的，因此下面的内容不仅适用于客户端计算机，同样也适用于服务器、路由器等各种设备[①]。

此外，以太网和 IP 一样，并不关心网络包的实际内容，因此以太网的收发操作也和 TCP 的工作阶段无关，都是共通的[②]。

2.5.7 将 IP 包转换成电或光信号发送出去

下面来看看以太网的包收发操作。IP 生成的网络包只是存放在内存中的一串数字信息，没有办法直接发送给对方。因此，我们需要将数字信息转换为电或光信号，才能在网线上传输，也就是说，这才是真正的数据发送过程。

负责执行这一操作的是网卡，但网卡也无法单独工作，要控制网卡还需要网卡驱动程序。驱动程序不只有网卡才有，键盘、鼠标、显卡、声卡等各种硬件设备都有。当然，不同厂商和型号的网卡在结构上有所不同，因此网卡驱动程序也是厂商开发的专用程序[③]。

网卡的内部结构如图 2.23 所示，这是一张网卡主要构成要素的概念图，并不代表硬件的实际结构[④]，但依然可以看清大体的思路。记住这一内部结构之后，我们再来介绍包收发的操作过程，现在，我们先来讲讲网卡的初始化过程。

网卡并不是通上电之后就可以马上开始工作的，而是和其他硬件一样，都需要进行初始化。也就是说，打开计算机启动操作系统的时候，网卡驱动程序会对硬件进行初始化操作，然后硬件才进入可以使用的状态。这些操作包括硬件错误检查、初始设置等步骤，这些步骤对于很多其他硬件也是共通的，但也有一些操作是以太网特有的，那就是在控制以太网收发操

① 路由器等网络设备的网卡是集成在设备内部的，其电路的设计也有所不同，尽管结构有差异，但功能和行为是没有区别的。

② 也和应用程序的种类无关。

③ 主要厂商的网卡驱动程序已经内置在操作系统中了。

④ 实际的内部结构随厂商和型号的不同而不同。

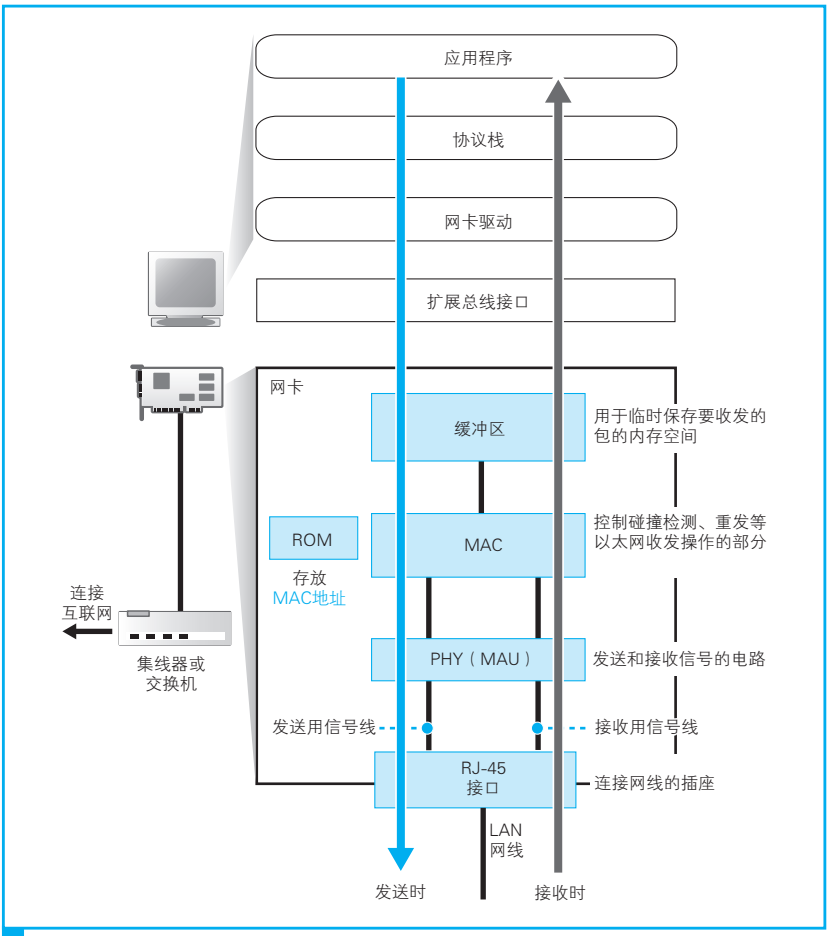


图 2.23　网卡

作的 MAC[①] 模块中设置 MAC 地址。

网卡的 ROM 中保存着全世界唯一的 MAC 地址，这是在生产网卡时

① MAC：Media Access Control 的缩写。MAC 头部、MAC 地址中的 MAC 也是这个意思。也就是说，通过 MAC 模块控制包收发操作时所使用的头部和地址就叫作 MAC 头部和 MAC 地址。

写入的，将这个值读出之后就可以对 MAC 模块进行设置，MAC 模块就知道自己对应的 MAC 地址了。也有一些特殊的方法，比如从命令或者配置文件中读取 MAC 地址并分配给 MAC 模块[①]。这种情况下，网卡会忽略 ROM 中的 MAC 地址。有人认为在网卡通电之后，ROM 中的 MAC 地址就自动生效了，其实不然，真正生效的是网卡驱动进行初始化时在 MAC 模块中设置的那个 MAC 地址[②]。在操作系统启动并完成这些初始化操作之后，网卡就可以等待来自 IP 的委托了。

网卡的 ROM 中保存着全世界唯一的 MAC 地址，这是在生产网卡时写入的。

网卡中保存的 MAC 地址会由网卡驱动程序读取并分配给 MAC 模块。

2.5.8　给网络包再加 3 个控制数据

好了，下面来看一看网卡是如何将包转换成电信号并发送到网线中的。网卡驱动从 IP 模块获取包之后，会将其复制到网卡内的缓冲区中，然后向 MAC 模块发送发送包的命令。接下来就轮到 MAC 模块进行工作了。

首先，MAC 模块会将包从缓冲区中取出，并在开头加上报头和起始帧分界符，在末尾加上用于检测错误的帧校验序列（图 2.24）[③]。

① 有些网卡驱动程序中不提供通过命令或配置文件设置 MAC 地址的功能。

② 通过命令或配置文件设置 MAC 地址时，必须注意不能和网络中其他设备的 MAC 地址重复，否则网络将无法正常工作。

③ 制定以太网标准的组织 IEEE 出于历史原因使用了“帧”而不是“包”，因此在以太网术语中都是说“帧”，其实我们基本没必要讨论两者的区别，大家可以认为包和帧是一回事，只是说法不同罢了。

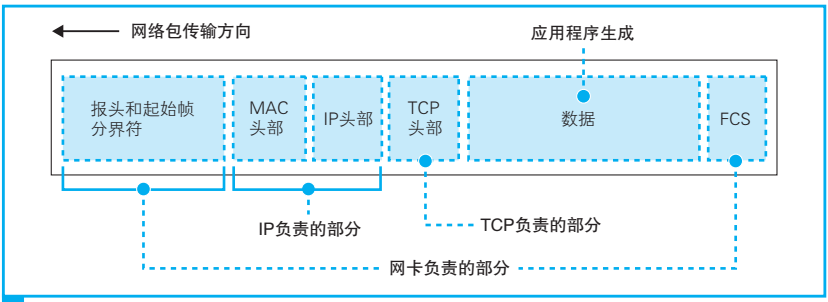


图 2.24　网卡发送出去的包

图中显示了协议栈和网卡对包的处理过程。MAC 头部很容易被误解为是由网卡来处理的，实际上它是由 TCP/IP 软件来负责的。

报头是一串像 10101010…这样 1 和 0 交替出现的比特序列，长度为 56 比特，它的作用是确定包的读取时机。当这些 1010 的比特序列被转换成电信号后，会形成如图 2.25 这样的波形。接收方在收到信号时，遇到这样的波形就可以判断读取数据的时机。关于这一块内容，我们得先讲讲如何通过电信号来读取数据。

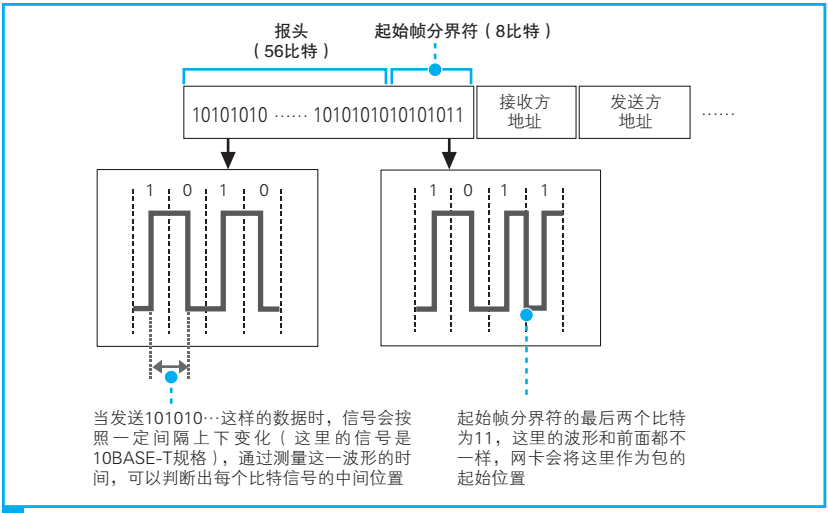
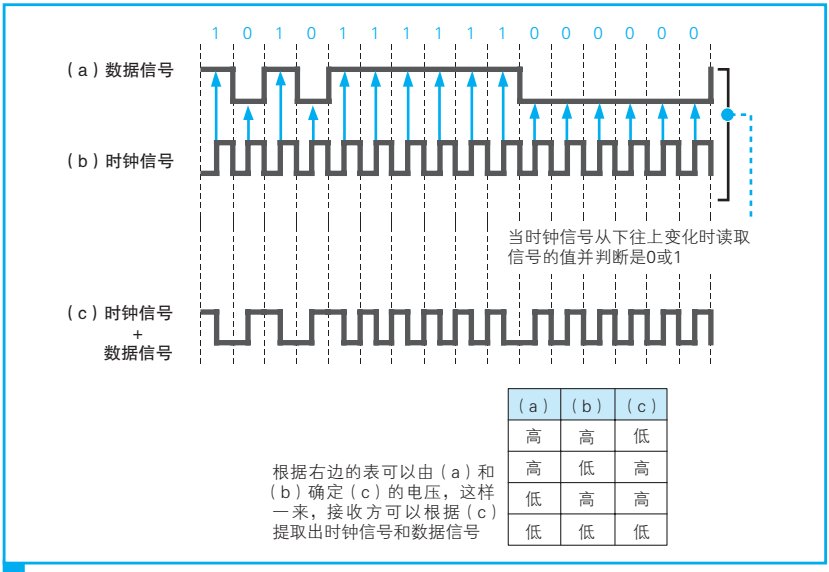


图 2.25　报头和起始帧分界符

每个包的前面都有报头和起始帧分界符（SFD），报头用来测定时机，SFD 用来确定帧的起始位置。

用电信号来表达数字信息时，我们需要让 0 和 1 两种比特分别对应特定的电压和电流，例如图 2.26（a）这样的电信号就可以表达数字信息。通过电信号来读取数据的过程就是将这种对应关系颠倒过来。也就是说，通过测量信号中的电压和电流变化，还原出 0 和 1 两种比特的值。然而，实际的信号并不像图 2.26 所示的那样有分隔每个比特的辅助线，因此在测量电压和电流时必须先判断出每个比特的界限在哪里。但是，像图 2.26（a）右边这种 1 和 0 连续出现的信号，由于电压和电流没有变化，我们就没办法判断出其中每个比特到底应该从哪里去切分。



（a）	（b）	（c）
高	高	低
高	低	高
低	高	高
低	低	低

图 2.26　通过时钟测量读取信号的时机
当信号连续为 1 或连续为 0 时，比特之间的界限就会消失，如果将时钟信号叠加进去，就可以判断出比特之间的界限了。

要解决这个问题，最简单的方法就是在数据信号之外再发送一组用来区分比特间隔的时钟信号。如图 2.26（b）所示，当时钟信号从下往上变化时[①]读取电压和电流的值，然后和 0 或 1 进行对应就可以了。但是这种方法

① 另外一种方法是当时钟信号从上往下变化时进行读取。

存在问题。当距离较远，网线较长时，两条线路的长度会发生差异，数据信号和时钟信号的传输会产生时间差，时钟就会发生偏移。

要解决这个问题，可以采用将数据信号和时钟信号叠加在一起的方法。这样的信号如图 2.26（c）所示，发送方将这样的信号发给接收方。由于时钟信号是像图 2.26（b）这样按固定频率进行变化的，只要能够找到这个变化的周期，就可以从接收到的信号（c）中提取出时钟信号（b），进而通过接收信号（c）和时钟信号（b）计算出数据信号（a），这和发送方将数据信号和时钟信号进行叠加的过程正好相反。然后，只要根据时钟信号（b）的变化周期，我们就可以从数据信号（a）中读取相应的电压和电流值，并将其还原为 0 或 1 的比特了。

这里的重点在于如何判断时钟信号的变化周期。时钟信号是以 10 Mbit/s 或者 100 Mbit/s 这种固定频率进行变化的，就像我们乘坐自动扶梯一样，只要对信号进行一段时间的观察，就可以找到其变化的周期。因此，我们不能一开始就发送包的数据，而是要在前面加上一段用来测量时钟信号的特殊信号，这就是报头的作用[①]。

以太网根据速率和网线类型的不同分为多种派生方式，每种方式的信号形态也有差异，并不都是像本例中讲的这样，单纯通过电压和电流来表达 0 和 1 的。因此，101010…这样的报头数字信息在转换成电信号后，其波形也不一定都是图 2.25 中的那个样子，而是根据方式的不同而不同。但是，报头的作用和基本思路是一致的。

报头后面的起始帧分界符在图 2.25 中也已经画出来了，它的末尾比特排列有少许变化。接收方以这一变化作为标记，从这里开始提取网络包数据。也就是说，起始帧分界符是一个用来表示包起始位置的标记。

末尾的 FCS（帧校验序列）用来检查包传输过程中因噪声导致的波形紊

① 如果在包信号结束之后，继续传输时钟信号，就可以保持时钟同步的状态，下一个包就无需重新进行同步。有些通信方式采用了这样的设计，但以太网的包结束之后时钟信号也跟着结束了，没有通过这种方式来保持时钟同步，因此需要在每个包的前面加上报头，用来进行时钟同步。

乱、数据错误，它是一串 32 比特的序列，是通过一个公式对包中从头到尾的所有内容进行计算而得出来的。具体的计算公式在此省略，它和磁盘等设备中使用的 CRC[①] 错误校验码是同一种东西，当原始数据中某一个比特发生变化时，计算出来的结果就会发生变化。在包传输过程中，如果受到噪声的干扰而导致其中的数据发生了变化，那么接收方计算出的 FCS 和发送方计算出的 FCS 就会不同，这样我们就可以判断出数据有没有错误。

2.5.9　向集线器发送网络包

加上报头、起始帧分界符和 FCS 之后，我们就可以将包通过网线发送出去了（图 2.24）。发送信号的操作分为两种，一种是使用集线器的半双工模式，另一种是使用交换机的全双工[②] 模式。

在半双工模式中，为了避免信号碰撞，首先要判断网线中是否存在其他设备发送的信号。如果有，则需要等待该信号传输完毕，因为如果在有信号时再发送一组信号，两组信号就会发生碰撞。当之前的信号传输完毕，或者本来就没有信号在传输的情况下，我们就可以开始发送信号了。首先，MAC 模块从报头开始将数字信息按每个比特转换成电信号，然后由 PHY，或者叫 MAU 的信号收发模块发送出去[③]。在这里，将数字信息转换为电信号的速率就是网络的传输速率，例如每秒将 10 Mbit 的数字信息转换为电信号发送出去，则速率就是 10 Mbit/s。

接下来，PHY（MAU）模块会将信号转换为可在网线上传输的格式，并通过网线发送出去。以太网规格中对不同的网线类型和速率以及其对应的信号格式进行了规定，但 MAC 模块并不关心这些区别，而是将可转换

① CRC：Cyclic Redundancy Check，循环冗余校验。

② 发送和接收同时并行的方式叫作“全双工”，相对地，某一时刻只能进行发送或接收其中一种操作的叫作“半双工”。

③ 根据以太网信号方式的不同，有些地方叫 MAU（Medium Attachment Unit，介质连接单元），有些地方叫 PHY（Physical Layer Device，物理层装置）。在速率为 100 Mbit/s 以上的以太网中都叫 PHY。

为任意格式的通用信号发送给 PHY（MAU）模块，然后 PHY（MAU）模块再将其转换为可在网线上传输的格式。大家可以认为 PHY（MAU）模块的功能就是对 MAC 模块产生的信号进行格式转换。当然，以太网还有很多不同的派生方式，网线传输的信号格式也有各种变化。此外，实际在网线中传输的信号很复杂，我们无法一一介绍，但是如果一点都不讲，大家可能对此难以形成一个概念，所以就举一个例子，大家感受一下就好[①]。图 2.27 就是这样一个例子，我们这里就不详细解释了，总之，网线中实际传输的信号就是这个样子的。

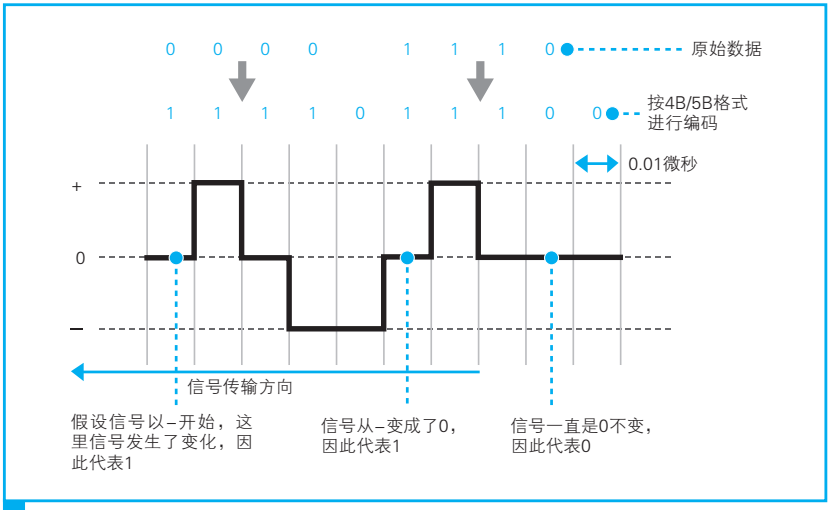


图 2.27　100BASE-TX 的信号

网卡的 MAC 模块生成通用信号，然后由 PHY（MAU）模块转换成可在网线中传输的格式，并通过网线发送出去。

① 图 2.26（c）中的数据信号和时钟信号叠加而成的信号，就是 10BASE-T 方式所使用的信号，这也是一个网线中实际传输的信号的例子。

PHY（MAU）的职责并不是仅仅是将 MAC 模块传递过来的信号通过网线发送出去，它还需要监控接收线路中有没有信号进来。在开始发送信号之前，需要先确认没有其他信号进来，这时才能开始发送。如果在信号开始发送到结束发送的这段时间内一直没有其他信号进来，发送操作就成功完成了。以太网不会确认发送的信号对方有没有收到。根据以太网的规格，两台设备之间的网线不能超过 100 米[①]，在这个距离内极少会发生错误，万一[②]发生错误，协议栈的 TCP 也会负责搞定，因此在发送信号时没有必要检查错误。

在发送信号的过程中，接收线路不应该有信号进来，但情况并不总是尽如人意，有很小的可能性出现多台设备同时进行发送操作的情况。如果有其他设备同时发送信号，这些信号就会通过接收线路传进来。

在使用集线器的半双工模式中，一旦发生这种情况，两组信号就会相互叠加，无法彼此区分出来，这就是所谓的信号碰撞。这种情况下，继续发送信号是没有意义的，因此发送操作会终止。为了通知其他设备当前线路已发生碰撞，还会发送一段时间的阻塞信号[③]，然后所有的发送操作会全部停止。

等待一段时间之后，网络中的设备会尝试重新发送信号。但如果所有设备的等待时间都相同，那肯定还会发生碰撞，因此必须让等待的时间相互错开。具体来说，等待时间是根据 MAC 地址生成一个随机数计算出来的。

当网络拥塞时，发生碰撞的可能性就会提高，重试发送的时候可能又会和另外一台设备的发送操作冲突，这时会将等待时间延长一倍，然后再次重试。以此类推，每次发生碰撞就将等待时间延长一倍，最多重试 10 次，如果还是不行就报告通信错误。

---

① 这是双绞线（twisted pair cable）的情况，如果采用光纤则可以更长，而且错误率不会上升。

② 实际的错误率低于万分之一，所以比“万一”还要小。

③ 阻塞信号：以太网中发生碰撞时，为了告知所有设备而发送的一种特殊信号。

另一种全双工模式我们会在第 3 章探索交换机时进行介绍，在全双工模式中，发送和接收可以同时进行，不会发生碰撞。因此，全双工模式中不需要像半双工模式这样考虑这么多复杂的问题，即便接收线路中有信号进来，也可以直接发送信号。

2.5.10 接收返回包

网卡将包转换为电信号并发送出去的过程到这里就结束了，既然讲到了以太网的工作方式，那我们不妨继续看看接收网络包时的操作过程[①]。

在使用集线器的半双工模式以太网中，一台设备发送的信号会到达连接在集线器上的所有设备。这意味着无论是不是发给自己的信号都会通过接收线路传进来，因此接收操作的第一步就是不管三七二十一把这些信号全都收进来再说。

信号的开头是报头，通过报头的波形同步时钟，然后遇到起始帧分界符时开始将后面的信号转换成数字信息。这个操作和发送时是相反的，即 PHY（MAU）模块先开始工作，然后再轮到 MAC 模块。首先，PHY（MAU）模块会将信号转换成通用格式并发送给 MAC 模块，MAC 模块再从头开始将信号转换为数字信息，并存放到缓冲区中。当到达信号的末尾时，还需要检查 FCS。具体来说，就是将从包开头到结尾的所有比特套用到公式中计算出 FCS，然后和包末尾的 FCS 进行对比，正常情况下两者应该是一致的，如果中途受到噪声干扰而导致波形发生紊乱，则两者的值会产生差异，这时这个包就会被当作错误包而被丢弃。

如果 FCS 校验没有问题，接下来就要看一下 MAC 头部中接收方 MAC 地址与网卡在初始化时分配给自己的 MAC 地址是否一致，以判断这个包是不是发给自己的。我们没必要去接收发给别人的包，因此如果不是自己的包就直接丢弃，如果接收方 MAC 地址和自己 MAC 地址一致，则

① 以太网的包接收操作和发送一样，和设备类型、TCP 的工作阶段以及应用程序的种类无关，都是共通的。