计算机网络

写在前面

咳咳，本文主要是面向计算机考研的同学（主要是自己也要考...），或者是对网络是如何运作感兴趣的大小朋友

尽可能用最生动形象的语言来解释过程吧，当然，在大部分知识点前会附上作者觉得可以加深理解的视频，这些视频也是作者学习时的参考资料，视频来自网络，如有侵权，联系秒删

感谢 B站湖科大教书匠 以及 《图解TCP / IP》一书， 在我的学习过程中给予莫大帮助

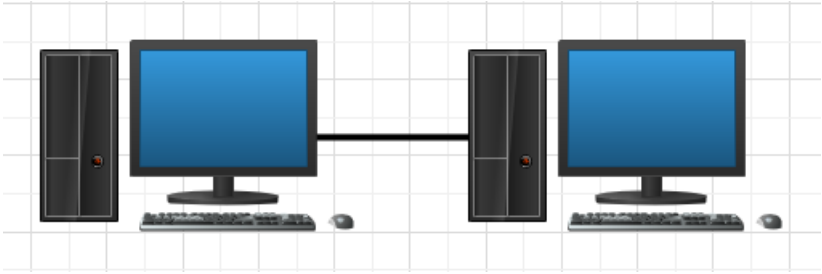
网络交换方式引入

我们首先从最简单的开始说起，首先我们使用网络的根本时希望计算机之间能够相互通信

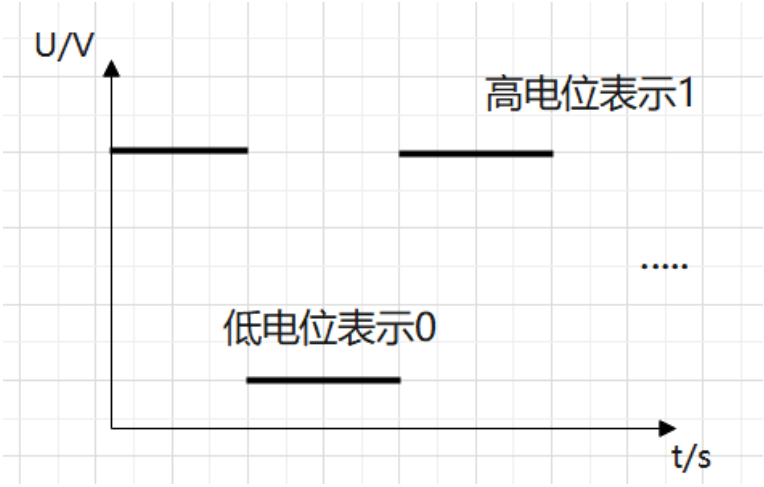
所以我们先从最简单的开始说起：

假设我们现在有且仅有两台计算机，我们可以采用的最简单的通信方式是什么？

毫无疑问是直接找根线将两个计算机直接连在一起

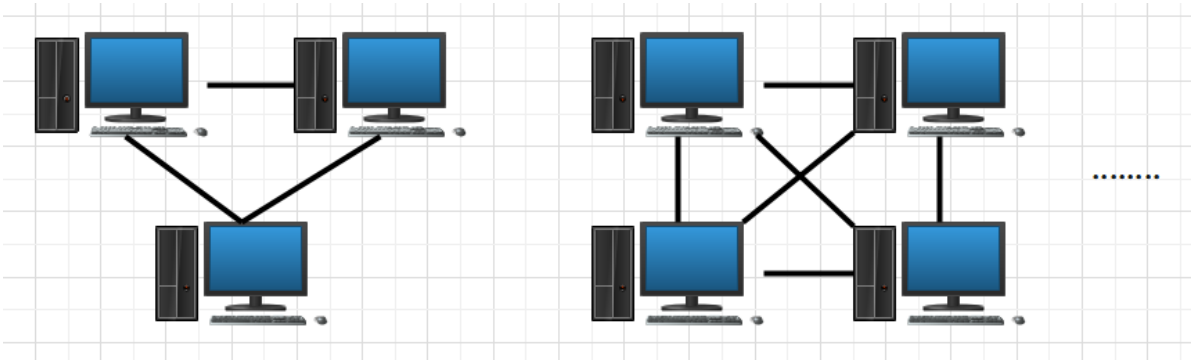


然后将传输的信息转化为 01 字符串，再通过高低电位来表示 01 字符串，这样就实现了信息在两台计算机之间的传输



接下来我们就将问题从两台计算机逐步提升到 $3, 4, 5, 6 \ldots$

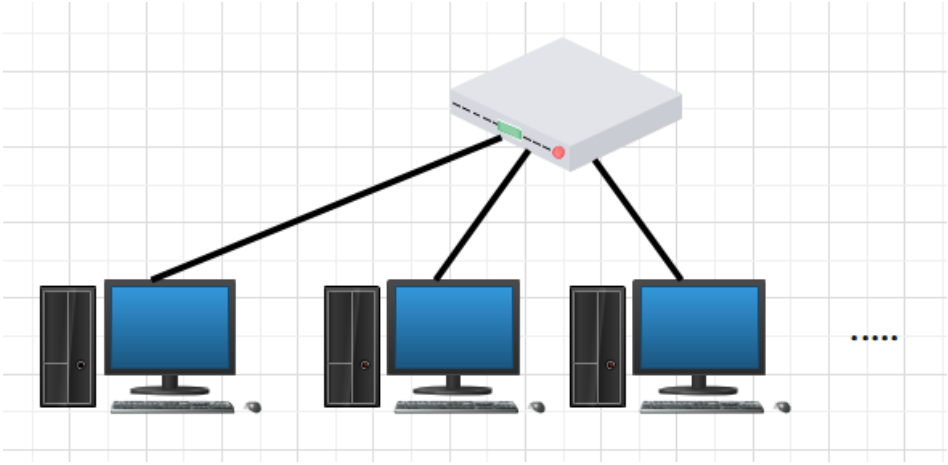
如果还是采用最朴素的方法，某个计算机与其他所有的计算机都有一根电线与其他计算机连接，就像下面这张图一样



这显然是不现实的，根据小学二年级的数学知识可得：

当一共有 n 台计算机两两相连， 就需要有 $\frac{n(n+1)}{2}$ 根电线

所以聪明的人民想到了其他的方法，也就是使用 “中间人” 帮忙交换的方法， 由 “中间人” 来帮忙与其他计算机进行通信

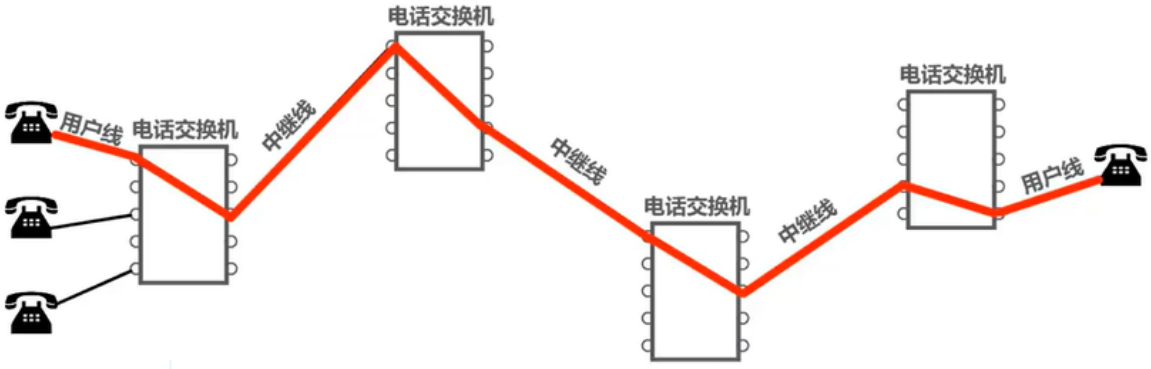


在这种思路下，存在以下三种交换方式，分别是 **电路交换** ， **报文交换** ， **分组交换**

电路交换、分组交换和报文交换（字幕版）

电路交换

电路交换是从前电话的通信方式，可以理解为当电路交换建立的时候，是存在一条线路直连两台设备的



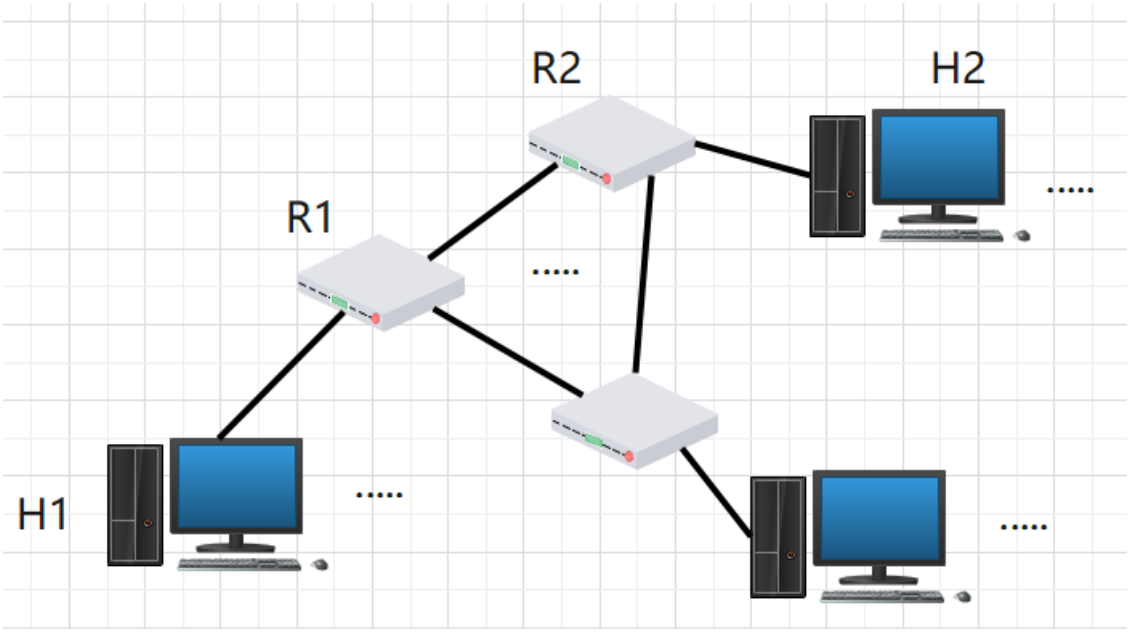
建立电路交换有三个步骤

- 建立连接 （分配通信资源）

  > 当通信双方都确认进行通信后， 就会形成一条专属于这个用户的电路通路

- 通信 （注意，这个时候会一直占用通信资源，其他用户无法使用这条为这两人建立的通路）
- 释放连接（归还通信资源，原本被占用的资源可以被别的用户使用）

报文交换



如图所示，假设 $H1$ 要发送信息给 $H2$， 那么它将在要发送的信息里面携带相关的目的地信息，接着将**全部信息**

发送给 $R1$， $R1$ 在接收到 **全部信息** 后将 **全部信息** 缓存在自身， 接着在发送给 $R2$， 最后发送给 $H2$

因为是发送 **全部信息**，所以要求这些 “中间人” 的缓存要足以容纳发送的信息

分组交换

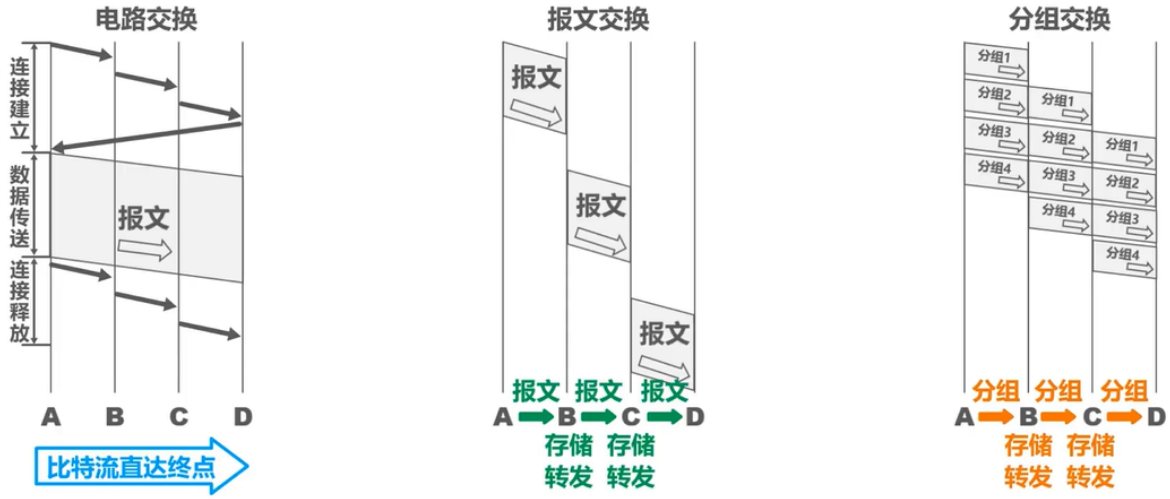
可以将分组交换看成是报文交换的升级版，这也是目前最常用的交换方法

注意到，之前的 “中间人” $R1, R2$ 需要将全部信息全部缓存下来，这有时候会造成不可避免的麻烦

所以在分组交换中，我们允许将信息切分成无数的小段， 并且在这些小段都会标注相应的序号，

这样哪怕是分散的消息也可以在接收端重新组装成完整的，正确顺序的消息

对比图



网络分层

如果你不是第一次接触网络， 那么你一定接触过OSI体系结构或者TCP/IP参考模型

不过为什么需要分层呢？ 在我最初学习的时候，也对这个问题有着疑问

首先要明白，**分层的最终目的是希望不同的电脑或者设备都能够相互互联**

但是不同的电脑或者设备其内部结构或者原理都有可能不同，所以就希望能寻找一种通用的方法来连接

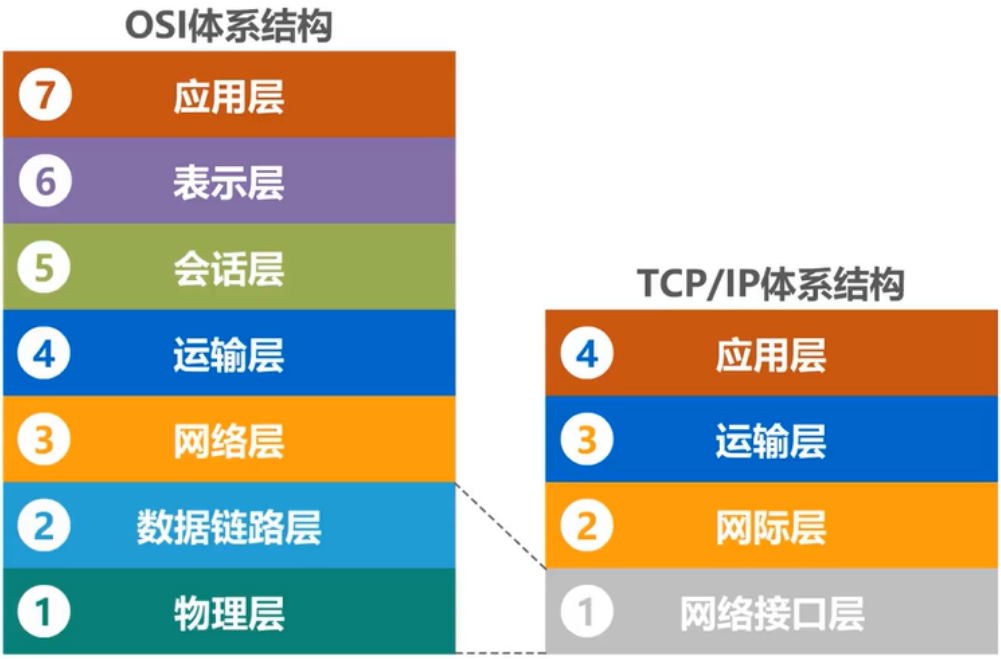
这种方法即是**协议**

> 可以将协议理解为日常生活中的语言，而不同的计算机可以理解为不同的国家
>
> 不同的国家之间如果想要相互交流，可以借助一种大家都知道，能听得懂得语言，套用在网络里即是协议

而不同得**分层是对不同协议的功能进行抽象**

用我们日常编程的习惯来说，我们更偏向于 高内聚低耦合，即希望可以将不同的功能进行模块化

这样对于维护以及更新都有着莫大的帮助



各分层的功能简述

> 物理层

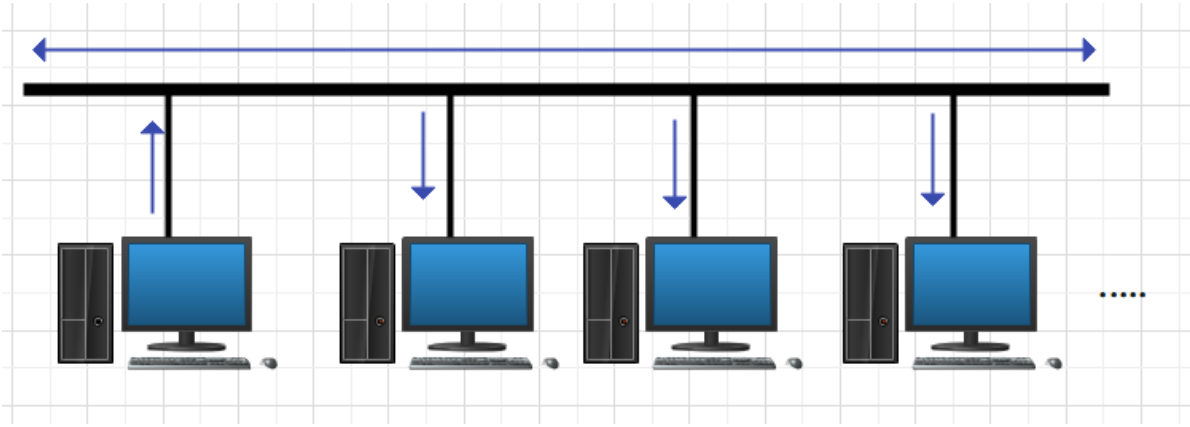
可以理解为是解决那些 "看得见" 的东西

- 采用什么传输介质 （双绞线， 无线传输....）
- 采用什么物理接口（HDMI， USB， Type-C ....）
- 使用什么信号来表示 01 （高低电位...）

这些都归物理层管

> 数据链路层

设想以下这个场景，对于同一个网络里面，或者说在如图所示的布局里：



第一台主机发送了一个消息，其他主机也一定会收到消息，那到底应该如何区分到底是谁在发消息，谁接收消息呢

这个问题就是数据链路层的所需要解决的问题

一个可行的方法是给所有的主机都编上一个唯一的主机标识，并且在要发送的信息上携带上

而这个唯一的主机标识就是网卡上的 **MAC地址**

不仅如此，数据链路层还需要管理如下问题

- 如何从发送的比特流里面识别处MAC地址

  > 即MAC地址携带的格式以及其他信息的格式

- 如果线路信息发生碰撞应该如何解决

  > 如果某一时刻有两台以上的主机发送消息，那么消息就会相互干扰，即信息碰撞

当然以上的总线式基本已被淘汰，现在多采用以太网交换机，但也不可避免的碰到这些问题

这一类问题的解决都是在数据链路层解决的

> **网络层**

注意之前所说的数据链路层是解决了在同一个网络的通信问题，但是如果是不同的网络该如何通讯

你可能会说，将全部网络看成一个网络不就可以了？但这样的话交换的效率会大大降低

而MAC地址是没有层级这一说的，也就是我仅有两个MAC地址是无法区分它们是否在同一个网络里面的

所以我需要对网络进行标注，即要同时标注网络以主机

这个问题就交给我们广为人知的 **IP 地址** 来解决了，它可以方便的确定是否在同一个网络

综上所述：确定不同网络的任务就交给了网络层

除此之外，网络层还需要解决以下问题

- 路由器如何进行分组转发
- 如何进行路由选择

> **运输层**

通常情况下，一台电脑上不止运行了一个程序，可能是浏览器，QQ，微信或者某些不可告人的软件

那么应该如何确定发送来的信息到底属于哪个应用程序呢?

即解决信息的归属问题， 解决这个问题的关键是 **端口号**

同时，万一信息在传输过程中发生了错乱，我们要如何校验呢，解决这个问题可以采用 **校验和**

总结来说， 运输层主要解决以下问题

- 解决进程之间的信息问题
- 遇到传输错误如何解决

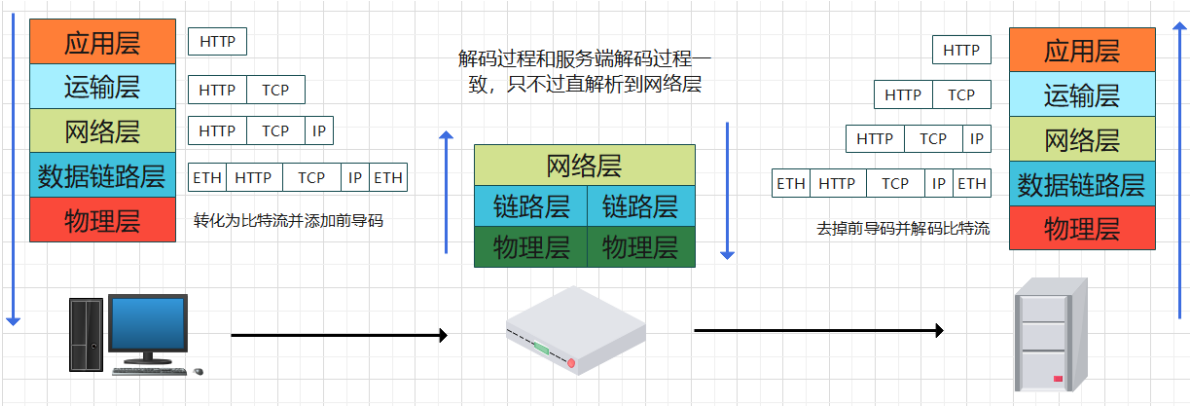
常见的协议有 **TCP 协议**， **UDP协议**

> **应用层**

..... 挖个坑

网络通信过程

图形化传输过程如下图所示：



文字描述：

> 发送端

- 应用层发出请求，封装成 HTTP报文， 这一层的数据统称为 **报文**
- 运输层将应用层的报文添加 TCP首部或者UDP首部，封装成 **报文段**
- 网络层给报文段添加了一个 IP 首部， 封装成 **IP数据报**
- 数据链路层将 IP数据包添加一个首部一个尾部，将其封装为 **帧**
- 物理层将发送的数据转化为比特流，并且添加前导码

> 接收端

对发送的内容逐层解码，最终获取到发送的数据

物理层

基本概念

物理层的主要目的是为了解决各种传输媒介传输比特流的问题

常见的传输介质有 **导引型传输介质** 和 **非导引型传输介质** （严格意义这一部分应该在物理层之下）

- 导引型传输介质

  > 双绞线　同轴电缆 光纤

- 非引导型传输介质

  > 微波通信

物理层协议的主要任务

- 机械特性

  > 指明接口所用接线器的形状和尺寸，即外观和物理细节

- 电气特性

> 接口电缆的各条线上的电压范围

- 功能特性

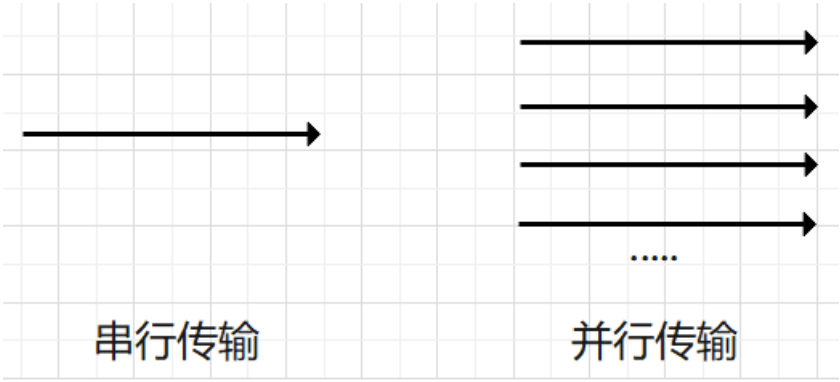
> 指明电压的意义

- 过程特性

> 指明对于不同的功能的可能出现事件的出现顺序

传输方式

- **串行传输**和**并行传输**



串行传输是每次只发送一个比特，所以只需要一条传输线路

并行传输是每次发送 n 个比特， 所以需要 n 条传输线路

在计算机内部的传输方式常采用串行传输，而计算机网络传输采用并行传输

- **单向通信**（单工）， **双向交替通信**（半双工）， **双向同时通信**（全双工）

单向通信（单工）： 数据是单向传输的，接收方不能发送消息 —— 收音机

双向交替通信（半双工）： 数据可以双向传播，但是不能同时进行 —— 对讲机

双向同时通信（全双工）：数据可以同时双向传播 —— 手机通话

编码与调制

首先我们先来了解以下数字信号和模拟信号

什么是模拟信号？数字信号？区别是什么？它们又是如何完成转换的？ <= 屑作者懒得自己写

接下来我们来介绍以下码元

所谓码元， 我们可以理解为一个周期，即一个码元表示这段时间内传输了一个 0 或者 1

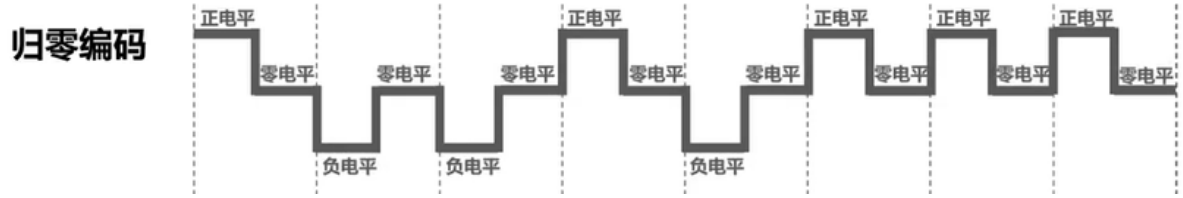
> 常用的编码方式

- 不归零编码

不过这种传输方式用一个致命缺点，就是当多个 0 或者 1 相连的时候，如何正确的识别数量

这就必须要求两台传输设备的时钟一致了，这会大大浪费资源

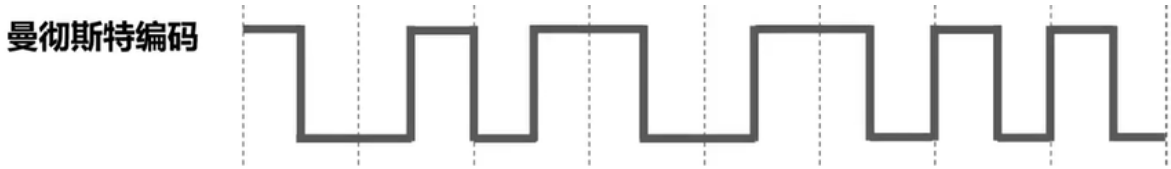
- 归零编码



在归零编码中，在码元的中间时刻设为零电平，这样就能清楚的分辨出数量

但这样做的缺点在于大部分数据带宽都用来归零了，被浪费掉了

- 曼彻斯顿编码



在每个码元的中间时刻都会发生跳变，我们可以根据跳变的规则来确定表示是 0 还是 1

例如： 如果是负跳变则是 0， 是正跳变则是 1

- 差分曼彻斯顿编码

注意这个时候，码元的中间时刻的跳变仅仅表示时钟

我们用是否发生跳变来表示传输的是 0 还是 1， 例如跳变是 0，不跳变是 1

相对于曼彻斯顿编码， 差分曼彻斯顿编码的变化更少，更适合较高的传输速率

> 调制

挖坑：)

信道容量极限

：)

数据链路层