China-pub.com
下载

第9章　二进制数

1973年，当安东尼·奥兰多在他写的一首歌中要求他挚爱的人“系一条黄色的绸带在橡树上”时，他并没有要求他的爱人进行繁琐的解释或冗长的讨论，只要求她给他一个简单的结果。他不去关心其中的因果，即使歌中复杂的感情和动情的历史在现实生活中重演，所有的人真正想知道的仅仅是一个简单的是或不是。他希望在树上系一条黄色的绸带来表示：“是的，即使你犯了很大的错，并且被判了入狱三年，我仍希望你回来和我一起共渡时光。”他希望用树上没有黄色的绸带来表示：“你连停在这里都别想。”

这是两个界线分明、相互排斥的答案。奥兰多没有这样唱：“如果你想再考虑一下的话，就系半条黄色的绸带”或者“如果你不爱我但仍希望我们是朋友，就系一条蓝色的绸带吧”。相反，他让答案非常的简单。

和黄色绸带的有无具有同样效果的另外几个例子（但可能无法用在诗里）是可以选择一种交通标记放在院外，可能是“请进”或“此路不通”。

或者在门上挂一个牌子，上写“关”或“开”。

或者用从窗口能够看到的一盏灯的亮灭来表示。

如果你只需说“是”或“不是”的话，可以有很多种方式来表达。你不必用一个句子来表达是或不是，也不需要一个单词，甚至连一个字母都不要。你只要用一个比特，即只要一个0或1即可。

正如我们在前面的章节中所了解到的，通常用来计数的十进制数事实上并没有什么与众不同的地方。非常清楚，我们的数字系统之所以是基于 10的（十进制数）是因为我们有 10个手指头。我们同样有理由使用八进制数字系统（如果我们是卡通人物）或四进制数字系统（如果我们是龙虾），甚至是二进制数字系统（如果我们是海豚）。

但是，二进制数字系统有一点儿特别：它可能是最简单的数字系统。二进制数字系统中只有两种二进制数字——0和1。要是我们想寻求更简单的数字系统，只好把数字 1去掉，这样，就只剩下0一个数字了。只有一个数字0的数字系统是什么都做不成的。

“bit(比特)”这个词被创造出来代表“binary digit”，它的确是新造的和计算机相关的最可爱的词之一。当然，“bit”有其通常的意义：“一小部分，程度很低或数量很少”。这个意义用来表示比特是非常精确的，因为1比特——一个二进制数字位——确实是一个非常小的量。

有时候当一个新词诞生时，它还包含了一种新的意思。 bit这个词也是这样。 1比特的意思超过了被海豚用来数数的二进制数字位所包含的意义。在计算机时代，比特已经被看作是组成信息块的基本单位。

当然，上述说法不一定完全正确，比特并不是传送信息的唯一的方式。字母、单词、摩尔斯码、布莱叶盲文，十进制数字都可以用来传递信息。比特传递的信息量很小。 1比特只具备最少的信息量，更复杂的信息需要多位比特来传递。（我们说比特传递的信息量小，并不是说它传送的信息不重要。事实上黄绸带对于与它相关的两个人来说是一个非常重要的信息。）

“听，孩子们，你们很快就能听到 Paul Revere 午夜的马蹄声。”亨利·朗费罗写道。尽管

China-pub.com
下载

他在描述Paul Revere是怎样通知美国人英国殖民者入侵的消息时不一定与史实完全一致，但他的确提供了一个利用比特传递信息的令人茅塞顿开的例子：

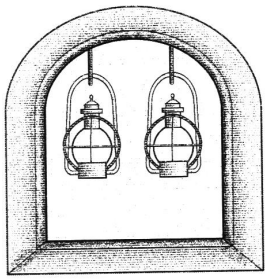
He said to his friend "If the British march
By land or sea from the town to-night,
Hang a lantern aloft in the belfry arch
Of the North Church tower as a special light,—
One, if by land, and two, if by sea..."

(他告诉他的朋友："如果英军今晚入侵，
你就在北教堂的钟楼拱门上悬挂点亮的提灯
作为信号。一盏提灯代表英军由陆路入侵，
两盏提灯代表英军由海路入侵。……)

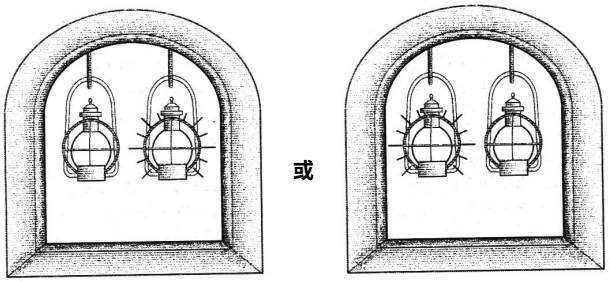
也就是说，Paul Revere 的朋友有两盏灯。如果英军由陆路入侵，他就挂一盏灯在教堂的钟楼上；如果英军由海路入侵，他就挂两盏灯在教堂的钟楼上。

然而，朗费罗并没有将所有的可能都涉及到。他留下第三种情况没有说，那就是英军根本就没有入侵的情况。朗费罗已经暗示第三种可能的信息可以由不挂提灯的方式来传递。

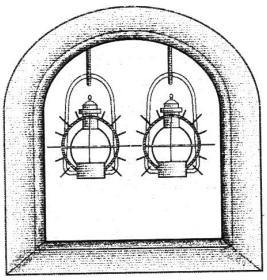
让我们假设那两盏灯是永久固定在教堂钟楼上的。在正常情况下，它们都不亮：



这就是指英军还没有入侵。如果一盏提灯亮：



表示英军正由陆路入侵。如果两盏提灯都亮：



China-pub.com 下载

表示英军正由海路入侵。

每一盏提灯都代表一个比特。亮着的灯表示比持值为 1，未亮的灯表示比特值为 0。前面奥兰多已经说明了传送只有两种可能性的信息只需要一个比特。如果 Paul Revere只需被告知英军正在入侵（不管是从何处入侵）的消息，一盏提灯就足够了。点亮提灯代表英军入侵，未点亮提灯代表又是一个和平之夜。

传递三种可能性的消息还需要再有一盏提灯。一旦再有一盏提灯，两个比特就可以通知有四种可能的信息：

00=英军今晚不会入侵
01=英军正由陆路入侵
10=英军正由陆路入侵
11=英军正由海路入侵

Paul Revere将三种可能性用两盏提灯来传送的做法事实上是相当富有经验的。用通信理论的术语说，他采用了冗余的办法来降低噪声的影响。通信理论中的噪声是指影响通信效果的任何事物。电话线路中的静电流显然是影响电话通信的一种噪声。然而，即使是在有噪声的情况下，电话通信仍能够成功，因为口语中存在大量的冗余。你同样可以听懂对方的话而无需将每个音节、每个字都听得很清楚。

在上述例子中，噪声是指晚上光线黯淡以及 Paul Revere距钟楼有一定的距离，它们都阻碍了Paul Revere声将钟楼上的两盏灯区分清楚。下面是朗费罗的诗中很重要的一段：

And lo! As he looks, on the belfry's height
A glimmer, and then a gleam of light!
He springs to the saddle, the bridle he turns,
But lingers and gazes, till full on his sight
A second lamp in the belfry burns!

（哦！他站在与钟楼等高的位置观察，
一丝微光，然后，有一盏灯亮了！
他跳上马鞍，调转马头，
徘徊，凝视，直到看清所有的灯
另一盏灯也亮了！）

那当然不是说Paul Revere正在辨清到底是哪盏灯先亮的问题。

这里最本质的概念是信息可能代表两种或多种可能性的一种。例如，当你和别人谈话时，说的每个字都是字典中所有字中的一个。如果给字典中所有的字从 1开始编号，我们就可能精确地使用数字进行交谈，而不使用单词。（当然，对话的两个人都需要一本已经给每个字编过号的字典以及足够的耐心。）

换句话说，任何可以转换成两种或多种可能的信息都可以用比特来表示。不用说，人类使用的很多信息都无法用离散的可能性来表示，但这些信息对我们人类的生存又是至关重要的。这就是人类无法和计算机建立起浪漫关系的原因所在（无论怎样，都希望这种情况不会发生）。如果无法将某些信息以语言、图片或声音的形式表达，那也不可能将这些信息以比特的形式编码。当然，你也不会想将它们编码。

举手或不举手是一个比特的信息。两个人是否举手 —— 就像电影评论家Roger Ebert和刚去

China-pub.com
下载

世不久的Gene Siskel对新影片提供他们最终的评价结果那样——传递两个比特的信息。（我们将忽略掉他们实际上对影片做的评语，而只关心他们有没有举手的问题。）这样，我们用两个比特代表四种可能：

00 = 他们都不喜欢这部影片
01 = Siskel 讨厌它，Ebert喜欢它
10 = Siskel 喜欢它，Ebert讨厌它
11 = Siskel和Ebert都喜欢它

第一个比特值代表Siskel的意见，0表示Siskel讨厌这部影片，1表示Siskel喜欢这部影片。同样，第二个比特值代表Ebert的意见。

因此，如果你的朋友问你Siskel和Ebert是怎么评价《Impolite Encounter》这部电影的，你不用回答“Siskel举手了，Ebert没有举手”或者“Siskel喜欢这部电影，Ebert不喜欢这部电影”，你可以简单地回答“么零”。你的朋友只要知道哪一位代表的是Siskel的意见，哪一位代表的是Ebert的意见，并且知道值为1代表举手，值为0代表没有举手，你的回答就是可以被人理解的。当然，你和你的朋友都要知道这种代码的含义。

我们也可以一开始就声明值为1的比特位表示没有举手，值为0的比特位表示举手了，这可能有点违反常规。通常我们会认为值为1的比特位代表正面的事情，而值为0的比特位代表相反的一方面，这的确只是一种很随意的指派。无论怎样，用此种代码的人只要明白0、1分别代表什么就可以了。

某一位或几位比特位的集合所代表的意义通常是和上下文相关的。橡树上的黄绸带可能只有系绸带的人和期望看到绸带的人知道其中的意思，改变绸带的颜色、系绸带的树或系绸带的日期，绸带可能会被认为只是一块毫无意义的破布。同样，要从Siskel和Ebert的手势中得到有用的信息，我们至少要知道正在讨论的是哪部影片。

如果你保存了Siskel和Ebert对一系列影片的评价和投票结果，你就有可能在表示Siskel和Ebert意见的比特信息中再增加一位代表你自己的观点的比特位。增加的第三位使得其代表的信息可能性增加到8种：

000 = Siskel讨厌它，Ebert讨厌它，我讨厌它
001 = Siskel讨厌它，Ebert讨厌它，我喜欢它
010 = Siskel讨厌它，Ebert喜欢它，我讨厌它
011 = Siskel讨厌它，Ebert喜欢它，我喜欢它
100 = Siskel喜欢它，Ebert讨厌它，我讨厌它
101 = Siskel喜欢它，Ebert讨厌它，我喜欢它
110 = Siskel喜欢它，Ebert喜欢它，我讨厌它
111 = Siskel喜欢它，Ebert喜欢它，我喜欢它

使用比特来表示信息的一个额外好处是我们清楚地知道我们解释了所有的可能性。我们知道有且仅有8种可能性，不多也不少。用3个比特，我们只能从0数到7，后面再没有3位二进制数了。

在描述Siskel和Ebert的比特时，你可能一直在考虑一个严重的，并且是令人烦恼的问题——对于Leonard Maltin的Movie &Video Guide怎么办呢？别忘了，Leonard Maltin是不采用举

China-pub.com 下载

手表决这种形式的，他对电影的评价用的是更传统的星级系统。

要想知道需多少个Maltin比特，首先要了解一些关于Maltin评分系统的知识。Maltin给电影的评价是1～4颗星，并且中间可以有半颗星。（仅仅是为了好玩，他实际上不会给电影只评一颗星，取而代之的是给一个BOMB［炸弹］。）这里总共有七种可能性，也就是说只需要3个比特位就可以表示一个特定的评价等级了：

000 = BOMB
001 = ★1/2
010 = ★★
011 = ★★1/2
100 = ★★★
101 = ★★★1/2
110 = ★★★★

你可能会问111怎么办呢，111这个代码什么意义都没有，它没有定义。如果二进制代码111被用来表示Maltin等级，那一定是出现错误了。（这可能是计算机出的错误，因为人不会给出这样的评分。）

前面我们曾用两个比特来代表Siskel和Ebert的评价结果，左边的一位代表Siskel的评价意见，右边的一位代表Ebert的评价意见。在上述Maltin评分系统中，各个比特位都有确定的意义吗？是的，当然有。将比特编码的数值加2再除以2，就得到了Maltin评分中对应的星的颗数。这样编码是由于我们在定义代码时遵循了合理性和连贯性，我们也可以下面的这种方式编码：

000=★★★
001=★1/2
010=★★1/2
011=★★★★
101=★★★1/2
110=★★
111=BOMB

只要大家都了解代码的含义，这种表示就和前述代码一样，都是合理的。

如果Maltin遇到了一部连一颗星都不值得给的电影，他就会给它半颗星。他当然有足够的代码来表示半颗星的情况，代码会像下面这样定义：

000=MAJOR BOMB
001=BOMB
010=★1/2
011=★★
100=★★1/2
101=★★★
110=★★★1/2
111=★★★★

China-pub.com
下载

但是，如果他再遇到连半颗星的级别都不够的影片并且决定给它没有星的级别（ATOMIC BOMB？），他就得再需要一个比特位了，已经没有3个比特的代码空闲了。

《Entertainment Weekly》杂志常常给事物定级，除了电影之外还有电视节目、CD、书籍、CD-ROM、网络站点等等。等级的范围从A+～F，如果你数一下的话，发现共有13个等级。这样，需要四个比特来代表这些等级：

0000 = F
0001 = D-
0010 = D
0011 = D+
0100 = C-
0101 = C
0110 = C+
0111 = B-
1000 = B
1001 = B+
1010 = A-
1011 = A
1100 = A+

有3个代码没有用到，它们是：1101、1110和1111，加上后总共是16个代码。

只要谈到比特，通常是指特定数目的比特位。拥有的比特位数越多，可以传递的不同可能性就越多。

对十进制数当然也是同样的道理。例如，电话号码的区号有几位呢？区号共有3位数字。如果所有的区号都使用的话（实际上有一部分区号并没有使用，将它们忽略），一共有10^3或1000个代码，从000～999。区号为212的7位数的电话号码有多少种可能呢？10^7或10 000 000个；区号为212并且以260开头的电话号码有多少个呢？104或10 000个。

同样，在二进制数中，可能的代码数等于2的比特位数次幂：

比特位数	代码数
1	$2^1 = 2$
2	$2^2 = 4$
3	$2^3 = 8$
4	$2^4 = 16$
5	$2^5 = 32$
6	$2^6 = 64$
7	$2^7 = 128$
8	$2^8 = 256$
9	$2^9 = 512$
10	$2^{10} = 1024$

每增加一个比特位，二进制代码数翻一番。

如果知道需要多少个代码，那么怎样才能知道需要多少个比特位呢？换句话说，在上述表中，如何才能由代码数反推出比特位数呢？

用到的方法叫作取以2为底的对数，对数运算是幂运算的逆运算。我们知道2的7次幂等于

China-pub.com 下载

128，以2为底的128的对数就等于7。用数学记号来表示第一个句子为：

$$2^7 = 128$$

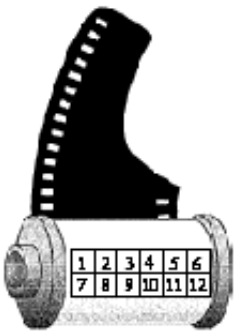
它与下述句子等价：

$$\log_2 128 = 7$$

因此，如果以2为底的128的对数等于7，以2为底的256的对数等于8，那么，以2为底的200的对数等于多少呢？大约是7.64，但实际上并不需要知道它。如果要表示200种不同的事物，我们共需要8个比特。

比特通常无法从日常观察中找到，它深藏于电子设备中。我们看不到压缩磁盘（CD）、数字手表或计算机中编过码的比特，但有时候比特也可以清晰地看到。

下面就是一个例子。如果你手头有一个使用35毫米胶片的相机，观察一下它的卷轴。这样拿住胶卷：



胶卷上有像国际跳棋棋盘一样的银色和黑色方格，方格已用数字1～12标识。这叫作DX编码，这12个方格实际上是12个比特。一个银色的方格代表值为1的比特，一个黑色的方格代表值为0的比特。方格1和7通常是银色的（代表1）。

这些比特是什么意思呢？你可能知道有些胶片对光的敏感程度要比其他胶片强，这种对光的敏感程度称作胶片速度。说对光非常敏感的胶片很快是因为这种胶片的曝光速度快。曝光速度是由ASA（American standards association，美国标准协会）来制定等级的，最常用的等级有100、200和400。ASA等级不只是以十进制数字的形式印在胶卷的外包装和暗盒上，而且还以比特的形式进行了编码。

胶卷总共有24个ASA等级，它们是：

25	32	40
50	64	80
100	125	160
200	250	320
400	500	640
800	1000	1250
1600	2000	2500
3200	4000	5000

为ASA等级编码需要多少个比特呢？答案是5个比特。我们知道，$2^4=16$，与24比太小了；$2^5 = 32$，又超过了所需的编码数。

China-pub.com
下载

比特值与胶片速度的对应关系如下所示：

方格2	方格3	方格4	方格5	方格6	胶片速度
0	0	0	1	0	25
0	0	0	0	1	32
0	0	0	1	1	40
1	0	0	1	0	50
1	0	0	0	1	64
1	0	0	1	1	80
0	1	0	1	0	100
0	1	0	0	1	125
0	1	0	1	1	160
1	1	0	1	0	200
1	1	0	0	1	250
1	1	0	1	1	320
0	0	1	1	0	400
0	0	1	0	1	500
0	0	1	1	1	640
1	0	1	1	0	800
1	0	1	0	1	1000
1	0	1	1	1	1250
0	1	1	1	0	1600
0	1	1	0	1	2000
0	1	1	1	1	2500
1	1	1	1	0	3200
1	1	1	0	1	4000
1	1	1	1	1	5000

多数现代的35毫米照相机胶片用的都是这些代码（除了那些要手工进行曝光的相机和具有内置式测光表但需要手工设置曝光速度的相机以外）。如果你看过照相机的内部放置胶卷的地方，你应该能够看到和胶片的金属方格（1～6号）相对应的6个金属可接触点。银色方格实际上是胶卷暗盒中的金属，是导体；油漆了的黑色方格，是绝缘体。

照相机的电子线路中有一支流向方格1的电流，方格1通常是银色的。这支电流有可能流到方格2～6，这要依方格中是纯银还是涂了油漆而定。这样，如果照相机在接触点4和5检测到了电流而在接触点2、3和6没有检测到，胶片的速度就是400ASA。照相机可以据此调节曝光时间。

廉价的照相机只要读方格2和方格3，并且假定胶片速度是50、100、200或400ASA四种可能速度之一。

多数相机不读方格8～12。方格8、9、10用来对这卷胶卷进行编码；方格11和12指出曝光范围，依胶片用于黑白照片、彩色照片还是幻灯片而定。

也许最常见的二进制数的表现形式是无处不在的UPC（universal product code，通用产品代码），即日常所购买的几乎所有商品包装上的条形码。条形码已经成为计算机在日常生活中应用的一种标志。

尽管UPC常常使人多疑，但它确实是一个无辜的小东西，发明出来仅仅是为了实现零售业的结算和存货管理的自动化，且其应用是相当成功的。当它和一个设计精良的结算系统共同使用时，顾客可以拿到列出细目的售货凭条，这一点是传统现金出纳员所无法做到的。

有趣的是，UPC也是二进制代码，尽管它初看起来并不像。将UPC解码并看看UPC码具

China-pub.com 下载

体是怎样工作的是很有益的。

通常情况下，UPC是30条不同宽度的垂直黑色条纹的集合，由不同宽度的间隙分割开，其下标有一些数字。例如，以下是Campbell公司$10\frac{3}{4}$盎司的罐装鸡汁面包装上的UPC：

可将条形码形象地看成是细条和黑条，窄间隙和宽间隙的排列形式，事实上，这是观察条形码的一种方式。黑色条有四种不同的宽度，较宽的条的宽度是最细条的宽度的两倍、三倍或者四倍。同样，各条之间的间隙中较宽的间隙是最窄间隙的两倍、三倍或者四倍。

但是，看待UPC的另一种方式是将它看作是一系列的比特。记住，整个条形码与条形码扫描仪在结算台“看”到的并不完全一样。扫描仪不会识别条形码底部的数字，因为识别数字需要一种更复杂的技术——光学字符识别技术，又称作OCR（optical character recognition）。实际上，扫描仪只识别整个条形码的一条窄带，条形码做得很大是为了便于结算台的操作人员用扫描仪对准顾客选购的物品。扫描仪所看到的那一条窄带可以这样表示：

它看上去是不是很像摩尔斯编码？

当计算机自左向右进行扫描时，它给自己遇到的第一个条分配一个值为1的比特值，给与条相邻的间隙分配一个值为0的比特值。后续的间隙和条被当作一行中一系列比特中的1个、2个、3个还是4个比特读进计算机要依据条或间隙的宽度而定。扫描进来的条形码的比特形式很简单：

因此，整个UPC只是简单的由95个比特构成的一串。本例中，这些比特可以像下面这样分组：

10100011010110001001100100011010001101000110101010111001011001101101100100111011001101000100101

比特	意义
101	最左边的护线
0001101	左边的数字
0110001	
0011001	
0001101	
0001101	
0001101	
01010	中间的护线
1110010	右边的数字
1100110	
1101100	
1001110	
1100110	
1000100	
101	最右边的护线

China-pub.com
下载

起初的3个比特通常是101，这就是最左边的护线，它帮助计算机扫描仪定位。从护线中，扫描仪可以知道代表单个比特的条或间隙的宽度，否则，所有包装上的 UPC印刷大小都是一样的。

紧挨着最左边的护线是每组有 7个比特位的六组比特串，每一组是数字 0～9的编码之一，我们在后面将证明这一点。接着的是 5个比特的中间护线，此固定模式（总是 01010）是一种内置式的检错码。如果扫描仪在应当找到中间护线的地方没有找到它，扫描仪就认为那不是UPC。中间护线是防止条形码被窜改或错印的方法之一。

中间护线的后面仍是每组 7个比特的6组比特串。最后是最右边的护线，也总是 101。最后的最右护线使得 UPC反向扫描（也就是自右向左扫描）同正向扫描一样成为可能，这一点我们将在后面解释。

因而整个UPC对12个数字进行了编码。左边的 UPC包含了6个数字的编码，每个数字占有7个比特位。你可以用下表进行解码：

左边的编码

0001101=0	0110001=5
0011001=1	0101111=6
0010011=2	0111011=7
0111101=3	0110111=8
0100011=4	0001011=9

注意，每个7位代码都是以0开头，以1结尾的。如果扫描仪遇到了第一个比特位值为 1或最后一个比特位值为 0的情况，它就知道自己没有将 UPC正确地读入或者是条形码被窜改了。另外我们还注意到每个代码都仅有两组连续的值为 1的比特位，这就意味着每个数字对应着条形码中的两个竖条。

上表中的每个代码中都包含有奇数个值为 1的比特位，这也是用于检测差错和数据一致性的一种机制，称为奇偶校验。如果一组比特位中含有奇数个 1，就称之为奇校验；如果含有偶数个1，就称之为偶校验。这样看来，所有这些代码都拥有奇校验。

为了给UPS右边的7位一组的数字解码，可以采用下面的表格：

右边的编码

1110010=0	1001110=5
1100110=1	1010000=6
1101100=2	1000100=7
1000010=3	1001000=8
1011100=4	1110100=9

这些代码都是前述代码的补码或补数：凡是 1的地方都换成0，凡是0的地方都换成1。这些代码都是以1开始，以零结束，并且每组都有偶数个 1，称之为偶校验。

现在，可以对UCP进行解码了。借助前两个表格， Campbell公司$10\frac{3}{4}$盎司的罐装鸡汁面的包装上用UPC编码的12个数字是：

0 51000 01251 7

这个结果是令人失望的，正如你所看到的那样，它们和印在 UPC底部的数字完全相同。

China-pub.com 下载

（这样做是有意义的，因为由于某种原因，扫描仪可能无法识别条形码，收银员就可以手工将这些数字输进去。）我们还没有完成解码的全部任务，而且，我们也无法从中解码任何秘密信息。然而，关于UPC的解码工作已经没有了，那30个竖条已经变成了12个数字。

第一个数字（在这里是0）被称为数字系统字符，0的意思是说这是一个规范的UPC编码。如果是具有不同重量的货物的UPC（像肉类或其他商品），这个数字是2；订单、票券的UPC编码的第一个数字通常是5。

紧接着的5个数字是制造商代码。在上例中，51000是Campbell鸡汁面公司的代码。Campbell公司生产的所有产品都使用这个代码。再后面的5个数字（01251）是该公司的某种产品的编号，上例中是指$10\frac{3}{4}$盎司的罐装鸡汁面。别的公司的鸡汁面可能有不同的编号，且01251在另外一个公司可能是指一种完全不同的产品。

和通常的想法相反，UPC中没有包含该种产品的价格。产品的价格信息可以从商店中使用的与该扫描仪相联的计算机中检索互到。

最后的数字（这里是7）称作模校验字符，这个字符可用来进行另外一种错误检验。为了解释校验字符是怎样工作的，将前11个数字（是0 51000 01251）各用一个字母来代替：

A BCDEF GHIJK

然后，计算下式的值：

3×（A+C+E+G+I+K）+（B+D+F+H+J）

从紧挨它并大于等于它的一个10的整倍数中减去它，其结果称为模校验字符。在上例中，有：

3×（0+1+0+0+2+1）+（5+0+0+1+5）=3×4+11=23

紧挨23并大于等于23的一个10的整倍数是30,故：

30－23=7

这就是印在外包装上并以UPC形式编码的模校验字符，这是一种冗余措施。如果扫描仪计算出来的模校验结果和UPC中编码中的校验字不一致，计算机就不能将这个UPC作为一个有效值接收。

正常情况下，表示从0～9的十进制数字只需4个比特就足够了。在UPC中，每个数字用了7个比特，这样总共有95个比特来表示11个有用的十进制数字。事实上，UPC中还包括空白位置（相当于9个0比特），位于左、右护线的两侧。因而，总共有113个比特用来编码11个十进制数，平均每个十进制数所用超过了10个比特位！

正像我们所知道的那样，有部分冗余对于检错来讲是必要的。这种商品编码如果能够很容易地被顾客用粗头笔修改的话，这种代码措施也就难以发挥其作用了。

UPC编码可以从两个方向读，这一点是非常有益的。如果扫描仪解码的第一个数字是偶校验（即：每7位编码中共有偶数个1），扫描仪就知道它正在从右向左进行解码。计算机系统用下表对右边的数字解码：

逆向时右边数字的代码

0100111 = 0	0111001 = 5
0110011 = 1	0000101 = 6
0011011 = 2	0010001 = 7
0100001 = 3	0001001 = 8
0011101 = 4	0010111 = 9

China-pub.com
下载

下面是对左边数字的解码表：

逆向时左边数字的代码

1011000 = 0	1000110 = 5
1001100 = 1	1111010 = 6
1100100 = 2	1101110 = 7
1011110 = 3	1110110 = 8
1100010 = 4	1101000 = 9

这些7位编码与扫描仪由左向右扫描时所读到的编码完全不同，但不会有模棱两可的现象。

让我们再看看本书中提到的由点、划组成其间用空格分开的摩尔斯电码。摩尔斯电码看上去不像是由0和1组成的，但它确实是。

下面回忆一下摩尔斯电码的编码规则：划的长度等于点长度的三倍；单个的点或划之间用长度与点的长度相等的空格分开；单词内的各个字母之间用长度等于划的长度的空格分隔；各单词之间由长度等于两倍的划长度的空格分开。

为使分析更加简单，我们假设划的长度是点长度的两倍而不是 3倍。也就是说，一个点是一个值为1的比特位，一个划是两个值为 1的比特位，空格是值为0的比特位。

下面是第2章的摩尔斯电码的基本表：

A	·—	J	·———	S	···
B	—···	K	—·—	T	—
C	—·—·	L	·—··	U	··—
D	—··	M	——	V	···—
E	·	N	—·	W	·——
F	··—·	O	———	X	—··—
G	——·	P	·——·	Y	—·——
H	····	Q	——·—	Z	——··
I	··	R	·—·		

下面是将它转化为比特形式的结果：

A	101100	J	1011011011000	S	1010100
B	110101010000	K	110101100	T	1100
C	110110110100	L	101101010000	U	101011000
D	110101000	M	110110000	V	101010110000
E	100	N	110100	W	101101100
F	101011010000	O	110110110000	X	110101011000
G	110110100	P	101101101000	Y	110101101100
H	101010000	Q	110110101100	Z	110110101000
I	10000	R	10110100		

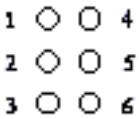
China-pub.com 下载

注意，所有的编码都以1开头，以两个0结束。结尾处的两个零代表单词中各个字母之间的空格，单词之间的空格用另外的一对0来表示。因而，“Hi,there”的摩尔斯电码通常是这样的：

但是，采用比特形式的摩尔斯电码看起来像UPC编码的横切面：

101010100101000011001010101001001011010010000

用比特的形式表示布莱叶盲文比表示摩尔斯电码容易得多。布莱叶编码是6比特代码。布莱叶盲文中的每一个字母都是由6个点组成的，点可能是凸起的，或没有凸起(平滑)的。如在第3章中讲的那样，这些点通常用数字1～6编号：



例如，单词“code”可以用布莱叶盲文这样表示：

如果突起的点是1，平坦的点是0，则布莱叶盲文中的每一个符号都可以用6个比特的二进制代码表示。单词“code”中的四个布莱叶字母符号就可以简单地写成：

100100 101010 100110 100010

最左边的一位对应编号为1的位置，最右边的一位对应编码为6的位置。

正如前面所讲到的，比特可以代表单词、图片、声音、音乐、电影，也可以代表产品编码、胶片速度、电影的受欢迎程度、英军的入侵以及某人所挚爱的人的意愿。但是，最基本的一点是：比特是数字。当用比特表示信息时只要将可能情况的数目数清楚就可以了，这样就决定了需要多少个比特位，从而使得各种可能的情况都能分配到一个编号。

比特在哲学和数学的奇怪混合物——逻辑——中发挥作用。逻辑最基本的目标是证明某个语句是否正确，正确与否也可以用1和0来表示。