数据转换器代码——您能解译这些代码吗?

作者：Walt Kester

简介

模数转换器(ADC)将模拟量——现实世界中绝大部分现象的特征——转换为数字语言，以便用于信息处理、计算、数据传输和控制系统。数模转换器(DAC)则用于将发送或存储的数据，或者数字处理的结果，再转换为现实世界的变量，以便控制、显示信息或进一步进行模拟处理。DAC和ADC的输入与输出之间的关系如图1所示。

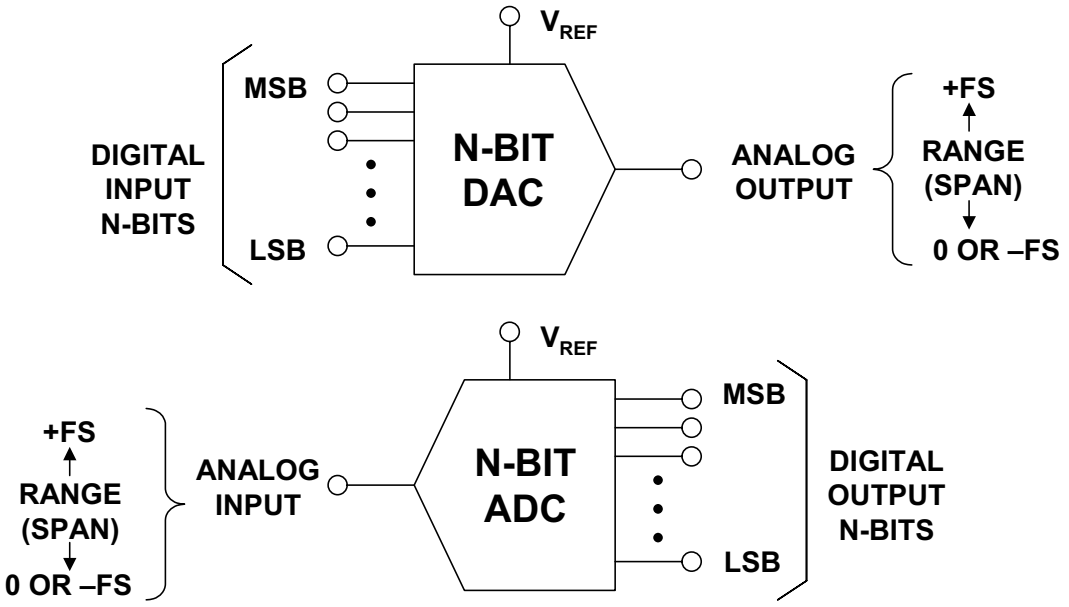

图1：数模转换器(DAC)和模数转换器(ADC)的输入和输出定义

无论其来源如何，模拟输入变量通常都是由传感器转换成电压或电流。这些电气量可以表现为以下形式：(1)在时域中对某一现象的快速或慢速直流连续直接测量；(2)经调制的交流波形（使用各种调制技术）；(3)或者某种组合形式，用相关变量的空间配置来代表轴角。第一种形式的例子有热电偶、直流基准源上的电位计和模拟运算电路的输出；“斩波”光学测量、交流应变计或电桥输出、噪声中嵌入的数字信号等属于第二种；自整角机和旋变器属于第三种。

本文讨论的模拟变量是那些用电压或电流来表示实际模拟现象的变量。它们可以是宽带，也可以是窄带；可以是直接测量的缩放形式，或者经过某种形式的模拟预处理，如线性化、组合、解调、滤波、采样保持等。

在处理过程中，电压和电流被“归一化”到与指定ADC输入范围兼容的范围。DAC的模拟输出电压或电流是归一化的直流信号，但随后可能会进行后处理（如调整、滤波、放大等）。

数字形式的信息一般用参考“地”的任意固定电压电平表示，或者出现在逻辑门的输出端，或者施加于其输入端。所用的数字值基本上是二进制，即每个“比特”（信息单位）有两种可能的状态：一种是“关”、“假”或“0”，另一种是“开”、“真”或“1”。这两种逻辑状态也可以用两种不同的电流水平来表示，但使用电流远不如使用电压普遍。另外，并不存在任何特殊原因要求电压必须参考地，发射极耦合逻辑(ECL)、正发射极耦合逻辑(PECL)和低压差分信号逻辑(LVDS)等就不是参考地。

“字”是一组表示数字值的电平，这些电平可以多种方式出现：*“并行”*同时出现在一条总线或一组门输入/输出上，*“串行”*（按照某一时间顺序）出现在一条线路上，或者作为一系列并行字节（即“字节串行”）/半字节（小字节）出现。例如，一个16位字可以占用一条16位总线的16位，也可以分为两个字节相继出现在一条8位总线上，或者分为四个4位半字节相继出现在一条4位总线上。

虽然有多种逻辑系统，但使用最广泛的是TTL（晶体管对晶体管逻辑），其中*“真”*或1对应于最小+2.4 V的输出电平（对于2.0 V以上的电平，输入明确地响应为1），*“假”*或0对应于最大+0.4 V的输出电平（对于+0.8 V以下的电平，输入明确地响应为0）。应当注意，如今虽然CMOS比TTL更受欢迎，但CMOS逻辑电平一般与较早的TTL逻辑标准兼容。

对于每个经量化的模拟电平，都会指定一组独特的并行或串行数字电平（或者称为*数值*、*代码*，表示模拟范围的一个独特部分）。一个典型的数字代码如下面的数组所示：

$$a_7\ a_6\ a_5\ a_4\ a_3\ a_2\ a_1\ a_0 = 1\ 0\ 1\ 1\ 1\ 0\ 0\ 1$$

它由8位组成，最左边的“1”称为最高有效位（MSB或第1位），最右边的“1”称为最低有效位（LSB或第*N*位，本例中为第8位）。该代码可能表示一个数值、一个字符或者一个模拟变量，要知道其确切含义，必须对*代码*和*转换关系*加以定义。注意，切勿混淆特定位的名称（即第1位、第2位等）与“a”数组的下标；下标对应于2的幂，它与序列中特定位的权重相关。

最为大家所熟悉的编码方式（十进制以外）是*自然或标准二进制*。二进制代码最常用于表示整数；在一个N位自然二进制整数代码中，LSB的权重为2^0（即1），下一位的权重为2^1（即2），依此类推直到MSB，其权重为2^{N-1}（即$2^N/2$）。二进制数的值是将所有非零位的权重相加而得。加权的位相加后，产生一个值在0到$2^N - 1$范围内的独特数字。每增加一个尾随零位（如有）,该数字的大小便加倍。

在转换器技术中，满量程（简称*FS*）与分辨率位数N无关。更有用的编码方式是*小数*二进制，它总是被归一化到满量程。整数二进制可以转译为小数二进制，方法是将所有整数值除以2^N。例如，MSB的权重为½（即$2^{(N-1)}/2^N = 2^{-1}$），下一位的权重为¼（即2^{-2}），依此类推直到LSB，其权重为$1/2^N$（即2^{-N}）。加权的位相加后，产生一个值在0到满量程的$(1 - 2^{-N})$范围内的2^N数字。附加位只会提供更精密的结构，而不影响满量程范围。十进制数与二进制数之间的关系以及各自的示例如图2所示。

WHOLE NUMBERS:

$$Number_{10} = a_{N-1}2^{N-1} + a_{N-2}2^{N-2} + \ldots + a_1 2^1 + a_0 2^0$$

MSB　　LSB

Example: $1011_2 = (1\times2^3) + (0\times2^2) + (1\times2^1) + (1\times2^0)$

$= 8 + 0 + 2 + 1 = 11_{10}$

FRACTIONAL NUMBERS:

$$Number_{10} = a_{N-1}2^{-1} + a_{N-2}2^{-2} + \ldots + a_1 2^{-(N-1)} + a_0 2^{-N}$$

MSB　　LSB

Example: $0.1011_2 = (1\times0.5) + (0\times0.25) + (1\times0.125) + (1\times0.0625)$

$= 0.5 + 0 + 0.125 + 0.0625 = 0.6875_{10}$

图2：用二进制数表示十进制数

单极性代码

在数据转换系统中，编码方法必须与ADC的模拟输入范围或DAC的模拟输出范围相关。最简单的情况是ADC的输入或DAC的输出始终为单极性正电压（DAC最常用的是电流输出，ADC则很少使用电流输入）。对于这类信号，最常用的编码方式是*标准二进制*，图3显示了一个4位转换器的例子。它有16种可能的电平，从全0代码0000到全1代码1111。必须注意，全1代码所代表的模拟值并不是满量程（简称FS），而是FS – 1 LSB。这是数据转换表示法的一个惯例，适用于ADC和DAC。图3给出了十进制等价数、二进制代码相对于满量程(FS)的值，以及各代码对应的电压水平（假设转换器满量程为+10 V）。图中还给出了等价格雷码，稍后将会讨论。

BASE 10 NUMBER	SCALE	+10V FS	BINARY	GRAY
+15	+FS – 1LSB = +15/16 FS	9.375	1 1 1 1	1 0 0 0
+14	+7/8 FS	8.750	1 1 1 0	1 0 0 1
+13	+13/16 FS	8.125	1 1 0 1	1 0 1 1
+12	+3/4 FS	7.500	1 1 0 0	1 0 1 0
+11	+11/16 FS	6.875	1 0 1 1	1 1 1 0
+10	+5/8 FS	6.250	1 0 1 0	1 1 1 1
+9	+9/16 FS	5.625	1 0 0 1	1 1 0 1
+8	+1/2 FS	5.000	1 0 0 0	1 1 0 0
+7	+7/16 FS	4.375	0 1 1 1	0 1 0 0
+6	+3/8 FS	3.750	0 1 1 0	0 1 0 1
+5	+5/16 FS	3.125	0 1 0 1	0 1 1 1
+4	+1/4 FS	2.500	0 1 0 0	0 1 1 0
+3	+3/16 FS	1.875	0 0 1 1	0 0 1 0
+2	+1/8 FS	1.250	0 0 1 0	0 0 1 1
+1	1LSB = +1/16 FS	0.625	0 0 0 1	0 0 0 1
0	0	0.000	0 0 0 0	0 0 0 0

图3：单极性二进制代码（4位转换器）

图4显示了一个采用标准二进制输入编码的理想3位DAC的传递函数。注意，全0输入代码对应的模拟输出为0。随着数字输入代码递增，模拟输出递增1 LSB/代码（本例中为1/8刻度）。最大输出电压为7/8 FS，对应FS – 1 LSB的值。当数字输入代码为100时，产生1/2 FS的中量程输出。

图5显示了一个理想3位ADC的传递函数。对于一定范围内的模拟输入电压，ADC都会产生给定的输出代码；该范围就是所谓*“量化不确定性”*，等于1 LSB。注意，对于理想ADC，相邻代码之间的过渡区宽度为0。但实际上，这些电平始终都有相关联的过渡噪声，因此宽度不为0。通常，用*“代码中心”*来定义给定代码对应的模拟输入，它位于两个相邻过渡区的中间位置（如图中的黑点所示）。这就要求第一过渡区出现在½ LSB处。满量程模拟输入电压定义为7/8 FS，即(FS – 1 LSB)。

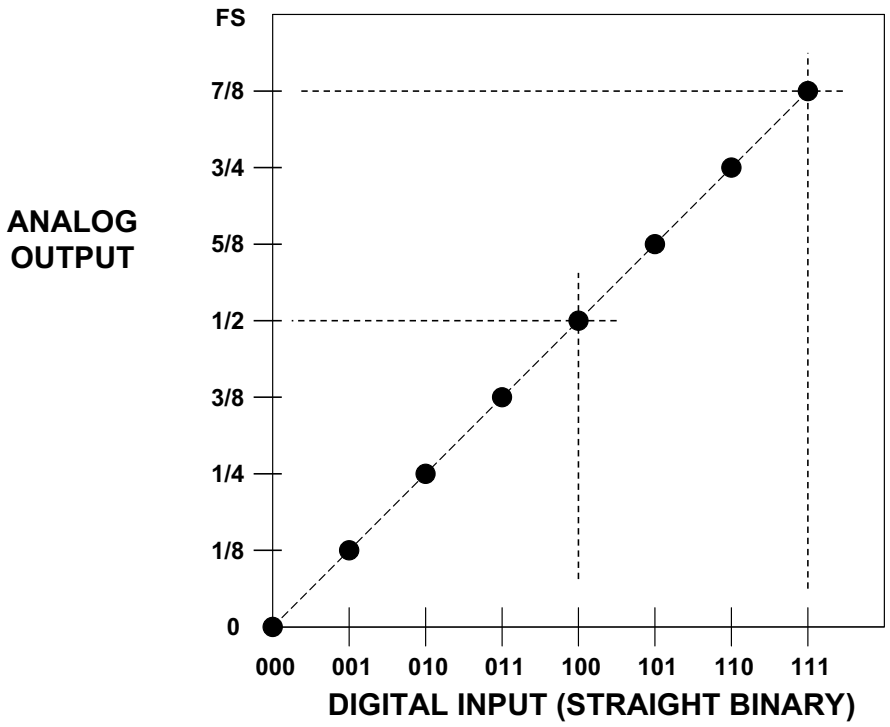

图4：理想单极性3位DAC的传递函数

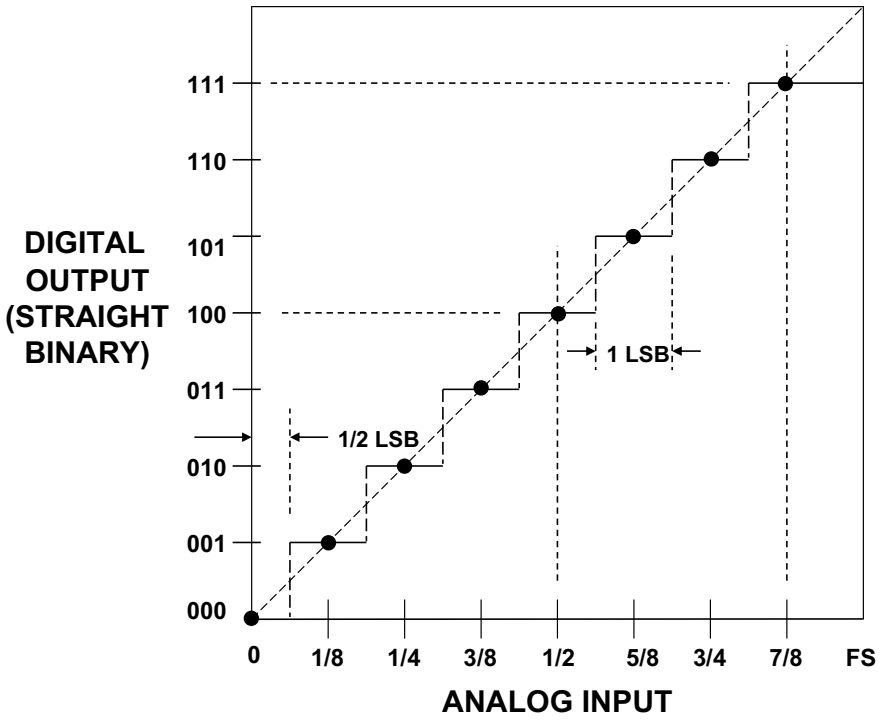

图5：理想单极性3位ADC的传递函数

格雷码

还有一种编码方式值得说明，它就是*格雷*码（或*反射二进制*），由Elisha Gray于1878年发明（参考文献1），后由Frank Gray于1949年重新发明（参考文献2）。图3最后一栏显示了4位标准二进制代码的等价格雷码。虽然它很少用于计算机运算，但它的一些有用特性对于模数转换很有吸引力。在格雷码中，当数值改变时，从一个代码跃迁到下一个代码只涉及到一位。二进制代码则不同，当从0111跃迁到1000时，所有位都要改变。某些ADC在内部使用格雷码，然后将格雷码转换为二进制代码供外部使用。

如上所述，采用格雷码时，相邻电平对应的格雷码字仅有一位之差。因此，如果针对特定电平的位判断有误差，则转换为二进制代码后的对应误差仅为1 LSB。对于中间电平，仅MSB改变。值得注意的是，基于比较器的现代Flash型转换器也可能由于比较器亚稳态而出现同样的现象。在少量过驱情况下，如果采用标准二进制解码技术，则比较器的输出可能会在其锁存输出中产生错误的判断，从而出现同样的现象。许多情况下，格雷码或伪格雷码用来解码比较器库。然后锁存格雷码输出，转换为二进制，并在最终输出时再次锁存。

轴编码器（角度转数字）和光学编码器等例子也常常在转换过程中使用格雷码来减小误差。

内部使用格雷码的ADC几乎都会将格雷码输出转换为二进制，供外部使用。利用图6所示的异或逻辑函数，很容易实现格雷码与二进制的互相转换。

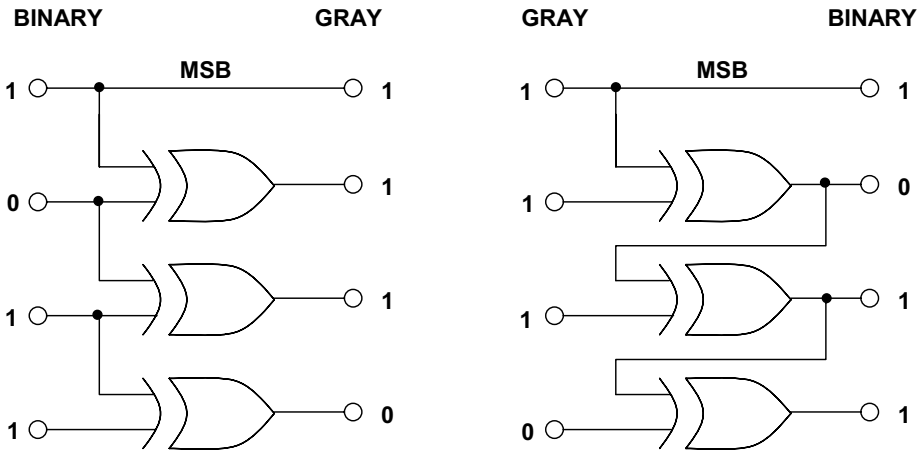

图6：利用异或逻辑函数实现二进制与格雷码的互相转换

双极性代码

在许多系统中，正负模拟量都需要用二进制代码来表示。*偏移二进制*、*二进制补码*、*二进制反码*或*符号幅度码*都可以实现这一点，但偏移二进制和二进制补码是目前最常用的编码方式。图7显示针对一个4位系统这两种编码的关系。注意，图中的值经过调整，以符合±5 V满量程输入/输出电压范围。

BASE 10 NUMBER	SCALE	±5V FS	OFFSET BINARY	TWOS COMP.	ONES COMP.	SIGN MAG.
+7	+FS – 1LSB = +7/8 FS	+4.375	1 1 1 1	0 1 1 1	0 1 1 1	0 1 1 1
+6	+3/4 FS	+3.750	1 1 1 0	0 1 1 0	0 1 1 0	0 1 1 0
+5	+5/8 FS	+3.125	1 1 0 1	0 1 0 1	0 1 0 1	0 1 0 1
+4	+1/2 FS	+2.500	1 1 0 0	0 1 0 0	0 1 0 0	0 1 0 0
+3	+3/8 FS	+1.875	1 0 1 1	0 0 1 1	0 0 1 1	0 0 1 1
+2	+1/4 FS	+1.250	1 0 1 0	0 0 1 0	0 0 1 0	0 0 1 0
+1	+1/8 FS	+0.625	1 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1
0	0	0.000	1 0 0 0	0 0 0 0	*0 0 0 0	*1 0 0 0
–1	– 1/8 FS	–0.625	0 1 1 1	1 1 1 1	1 1 1 0	1 0 0 1
–2	– 1/4 FS	–1.250	0 1 1 0	1 1 1 0	1 1 0 1	1 0 1 0
–3	– 3/8 FS	–1.875	0 1 0 1	1 1 0 1	1 1 0 0	1 0 1 1
–4	–1/2 FS	–2.500	0 1 0 0	1 1 0 0	1 0 1 1	1 1 0 0
–5	–5/8 FS	–3.125	0 0 1 1	1 0 1 1	1 0 1 0	1 1 0 1
–6	–3/4 FS	–3.750	0 0 1 0	1 0 1 0	1 0 0 1	1 1 1 0
–7	– FS + 1LSB = –7/8 FS	–4.375	0 0 0 1	1 0 0 1	1 0 0 0	1 1 1 1
–8	– FS	–5.000	0 0 0 0	1 0 0 0		

NOT NORMALLY USED
IN COMPUTATIONS (SEE TEXT)

*

	ONES COMP.	SIGN MAG.
0+	0 0 0 0	0 0 0 0
0–	1 1 1 1	1 0 0 0

图7：双极性代码（4位转换器）

对于*偏移二进制*，零电平信号值对应代码1000。代码顺序与标准二进制相同。标准二进制与偏移二进制系统的唯一区别在于模拟信号相关的半量程偏移不同。最小的负值(–FS + 1 LSB)对应代码0001，最大的正值(+FS – 1 LSB)对应代码1111。注意，为了保持关于中间电平的完美对称性，计算中一般不使用代表负满量程(–FS)的全0代码(0000)。它可以用来代表负超量程条件，或者简单地为其指定0001 (–FS + 1 LSB)的值。

一个双极性3位DAC的偏移二进制代码与模拟输出范围的关系如图8所示。对于0值输入代码100，DAC的模拟输出为0。最小的负输出电压一般对应于代码001 (–FS + 1 LSB)，最大的正输出电压一般对应于代码111 (+FS – 1 LSB)。需要时，可以使用000输入代码对应的输出电压，但这会使输出丧失关于0的对称性，数学计算将变得复杂。

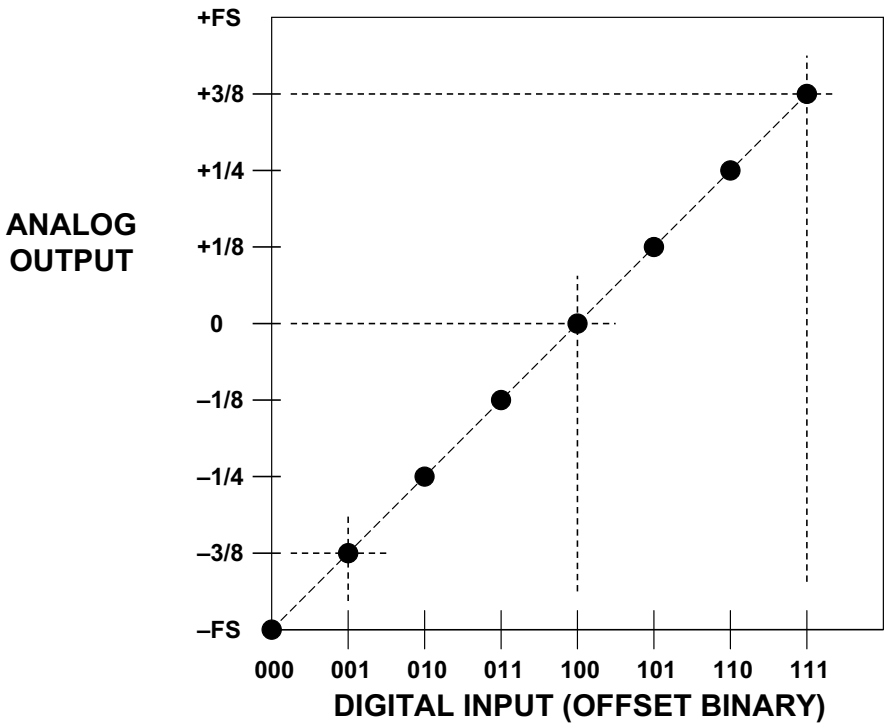

图8：理想双极性3位DAC的传递函数

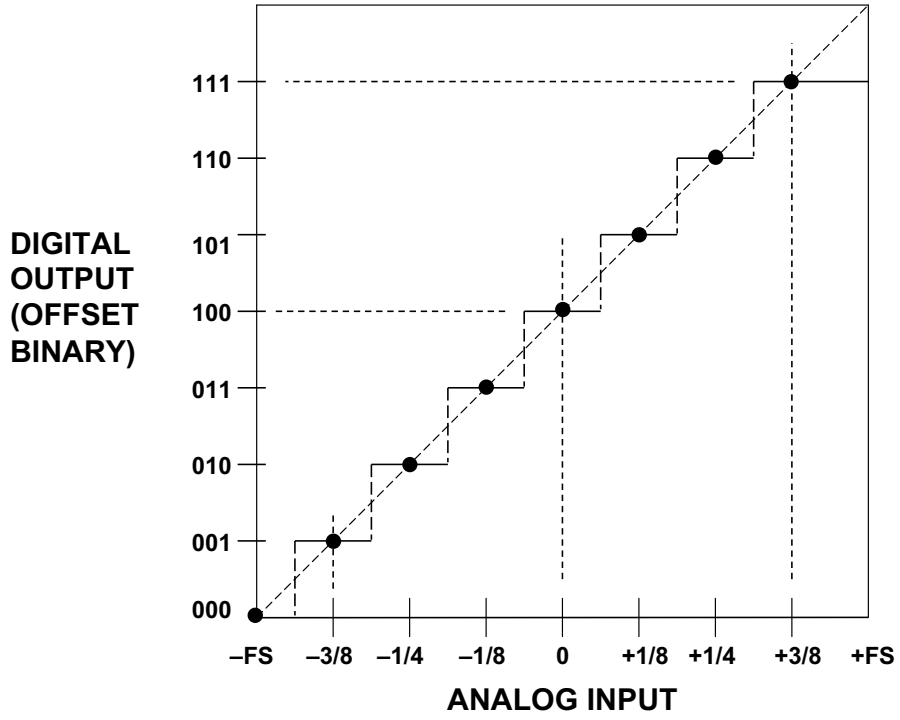

图9：理想双极性3位ADC的传递函数

一个双极性3位ADC的偏移二进制输出代码与其模拟输入的关系如图9所示。注意，模拟输入0对应中间电平代码100。像双极性DAC一样，最小的负输入电压一般对应于代码001 (–FS + 1 LSB)，最大的正输入电压一般对应于代码111 (+FS – 1 LSB)。如上所述，需要时，可以使用000输出代码，但这会使输出丧失关于0的对称性，数学计算将变得复杂。

*二进制补码*与MSB为补码（反转）的偏移二进制相同。显而易见，这在转换器中非常容易实现，只需使用一个变换器或取得D型正反器的互补输出。二进制补码编码方式之所以颇受欢迎，是因为利用它很容易在计算机和DSP中执行数学运算。就转换而言，二进制补码包括表示正幅度（0符号位）的二进制代码，各正数的二进制补码表示其负值。产生二进制补码的数学方法是求一个数的补数并加上1 LSB。例如，通过求取+3/8 FS的二进制补码，便可获得–3/8 FS。具体做法是：首先求取+3/8 FS对应的代码0011的补码，得到1100；然后加上1 LSB，得到1101。

二进制补码使减法运算变得容易。例如，若要从4/8 FS中减去3/8 FS，则将4/8与–3/8相加，即0100加上1101，结果为0001或1/8，不用考虑进位。

*二进制反码*也可以用来表示负数，但它远不如二进制补码那样受欢迎，如今鲜有使用。二进制反码的获得方法是求取正数所有位的补数。例如，3/8 FS (0011)的二进制反码是1100。各正值的补数的二进制反码表示其相应的负值。这包括0，表示0的有两个代码：0000（称为0+）或1111（称为0–）。这种非单值性必须通过数学方法消除，显然会给只有一个代码表示0的ADC和DAC带来问题。

*符号幅度码*似乎是用数字方式表示带符号模拟量的最简单方式，它只需确定与幅度相称的代码并加上一个极性位。符号幅度BCD在双极性数字电压表中很常用，但存在0对应两个代码的问题。因此，对于涉及到ADC或DAC的大多数应用，这种方法并不受欢迎。

图10总结了各种双极性代码（偏移二进制、二进制补码、二进制反码和符号幅度码）之间的关系，并给出了互转方法。

To Convert From → / To ↓	Sign Magnitude	2's Complement	Offset Binary	1's Complement
Sign Magnitude	No Change	If MSB = 1, complement other bits, add 00…01	Complement MSB If new MSB = 1, complement other bits, add 00…01	If MSB = 1, complement other bits
2's Complement	If MSB = 1, complement other bits, add 00…01	No Change	Complement MSB	If MSB = 1, add 00…01
Offset binary	Complement MSB If new MSB = 0 complement other bits, add 00…01	Complement MSB	No Change	Complement MSB If new MSB = 0, add 00…01
1's Complement	If MSB = 1, complement other bits	If MSB = 1, add 11…11	Complement MSB If new MSB = 1, add 11…11	No Change

图10：各种双极性代码之间的关系

本部分需要考虑的最后一种代码是*二进制编码的十进制码(BCD)*，它将十进制数的每个十进制位（0到9）表示为相应的4位标准二进制字，如图11所示。最小数字0表示为0000，数字9表示为1001。这种代码的效率相对较低，因为在每个十进制位的16种代码状态中，只有10种状态得到利用。然而，当与数字电压表等十进制显示器接口时，它是一种非常有用的代码。

BASE 10 NUMBER	SCALE	+10V FS	DECADE 1	DECADE 2	DECADE 3	DECADE 4
+15	+FS – 1LSB = +15/16 FS	9.375	1 0 0 1	0 0 1 1	0 1 1 1	0 1 0 1
+14	+7/8 FS	8.750	1 0 0 0	0 1 1 1	0 1 0 1	0 0 0 0
+13	+13/16 FS	8.125	1 0 0 0	0 0 0 1	0 0 1 0	0 1 0 1
+12	+3/4 FS	7.500	0 1 1 1	0 1 0 1	0 0 0 0	0 0 0 0
+11	+11/16 FS	6.875	0 1 1 0	1 0 0 0	0 1 1 1	0 1 0 1
+10	+5/8 FS	6.250	0 1 1 0	0 0 1 0	0 1 0 1	0 0 0 0
+9	+9/16 FS	5.625	0 1 0 1	0 1 1 0	0 0 1 0	0 1 0 1
+8	+1/2 FS	5.000	0 1 0 1	0 0 0 0	0 0 0 0	0 0 0 0
+7	+7/16 FS	4.375	0 1 0 0	0 0 1 1	0 1 1 1	0 1 0 1
+6	+3/8 FS	3.750	0 0 1 1	0 1 1 1	0 1 0 1	0 0 0 0
+5	+5/16 FS	3.125	0 0 1 1	0 0 0 1	0 0 1 0	0 1 0 1
+4	+1/4 FS	2.500	0 0 1 0	0 1 0 1	0 0 0 0	0 0 0 0
+3	+3/16 FS	1.875	0 0 0 1	1 0 0 0	0 1 1 1	0 1 0 1
+2	+1/8 FS	1.250	0 0 0 1	0 0 1 0	0 1 0 1	0 0 0 0
+1	1LSB = +1/16 FS	0.625	0 0 0 0	0 1 1 0	0 0 1 0	0 1 0 1
0	0	0.000	0 0 0 0	0 0 0 0	0 0 0 0	0 0 0 0

图11：二进制编码的十进制(BCD)代码

补码

某些形式的数据转换器（例如早期使用单片NPN四通道电流开关的DAC）要求自然二进制或BCD等标准代码，但所有位都用补码表示。这种代码称为*“补码”*。目前讨论的所有代码的补码都可以通过这种方法获得。注意不要混淆*补码*与*二进制反码*或*二进制补码*。

在一个4位补码二进制转换器中，0用1111表示，半量程用0111表示，FS – 1 LSB用0000表示。实践中，补码通常用寄存器的负输出获得，而不使用其正输出（寄存器可同时提供正负输出）。

某些情况下，可以利用补码来使DAC的模拟输出反相。如今有许多DAC提供差分输出，因而无需更改输入代码即可实现极性反转。同样，有许多ADC提供差分逻辑输出，利用这些输出也可以实现极性反转。

致谢

ADI公司同仁Dan Sheingold允许笔者直接引用其经典著作*《模数转换器手册》*（1986，参考文献3）中的内容，在此深表感谢。

参考文献：

1. K. W. Cattermole, *Principles of Pulse Code Modulation*, American Elsevier Publishing Company, Inc., 1969, New York NY, ISBN 444-19747-8. (An excellent tutorial and historical discussion of data conversion theory and practice, oriented towards PCM, but covers practically all aspects. This one is a must for anyone serious about data conversion!)

2. Frank Gray, "Pulse Code Communication," *U.S. Patent 2,632,058*, filed November 13, 1947, issued March 17, 1953. (detailed patent on the Gray code and its application to electron beam coders).

3. Dan Sheingold, *Analog-Digital Conversion Handbook, 3rd Edition*, Analog Devices and Prentice-Hall, 1986, ISBN-0-13-032848-0. (the defining and classic book on data conversion).