II Understanding Converters

Chapter II-1

A/D converters translate from analog measurements, which are characteristic of most phenomena in the "real world," to digital language, used in information processing, computing, data transmission, and control systems. D/A converters are used in transforming transmitted data or the results of computation back to "real-world" variables for control, information display, or further analog processing.

ANALOG QUANTITIES

Analog input variables, whatever their origin, are most frequently converted by transducers into voltages or currents. These electrical quantities may appear as "dc" continuous direct measurements of a phenomenon, as modulated ac waveforms (using a wide variety of modulation techniques), or in some combination, with a spatial configuration of related variables to represent shaft angles. Examples of the first are outputs of thermocouples, potentiometers on dc references, analog computers; of the second, "chopped" optical measurements, ac strain gage or bridge outputs; and of the third, synchros and resolvers.

The analog variables to be dealt with in this chapter are those involving "dc" voltages or currents representing the actual analog phenomena. They may be either wideband or narrow-band. They may be either scaled from the direct measurement, or subjected to some form of analog pre-processing, such as linearization, combination, demodulation, filtering, sample-hold, etc. As part of the process, the voltages and currents are "normalized" to ranges compatible with assigned converter input ranges. Ways and means of accomplishing appropriate pre-processing are discussed in the chapters on applications and system accessories. Analog output voltages or currents from D/A converters are direct and in normalized form, but they may be subsequently post-processed (e.g., scaled, filtered, boosted, etc.).

Synchro-to-digital and digital-to-synchro converters, while widely-used in some control applications, are not included within the scope of this chapter. Some relevant material on this topic will be found in the applications chapters of Part 1.

DIGITAL QUANTITIES

Digital numbers are represented by the presence or absence of essentially fixed voltage levels referred to "ground," either occurring at the outputs of logic gates, or applied to their inputs. The digital numbers used are all basically binary. That is, each "bit," or unit of information has one of two possible states. These states are "off," "false," or "0," and "on," "true," or "1." *Words*, groups of levels representing digital numbers, may appear simultaneously in *parallel*, on groups of gate inputs or outputs, *serially* (or in a time sequence) on a single line,[1] or in a serial-parallel combination (e.g. "*byte-serial*").

[1] In serial data transmission, if the levels return to ground between successive bits, they are denoted RZ (return-to-zero); if they change only when the leading edge of a clock pulse is present, and remain until the next leading edge, they are denoted NRZ (non-return-to-zero).

The most widely-used choice of levels at this writing, applicable to the class of products to which this book is devoted (i.e., converters manufactured by Analog Devices), are those used in TTL (transistor-transistor logic), in which positive true, or "1" corresponds to a minimum output level of +2.4V (inputs respond unequivocally to "1" for levels greater than 2.0V); and false, or "0" corresponds to a maximum output level of +0.4V (inputs respond unequivocally to "0" for anything less than +0.8V). A unique parallel or serial grouping of digital levels, or a *number*, or *code*, is assigned to each analog level which is quantized (i.e., represents a unique portion of the analog range). A typical digital code would be this array:

1 0 1 1 1 0 1 0 1

It is composed of nine bits. The "1" at the extreme left is called the "most significant bit" (MSB, or Bit 1), and the one at the right is called the "least significant bit" (LSB, or bit n: 9 in this case). The meaning of the code, either as a number, or as a representation of an analog variable, is unknown until the *code* and the *conversion relationship* have been defined.

THE BINARY CODE

The best-known code is *natural binary*. In a natural binary fractional code having n bits, the MSB has a weight of 1/2: (2^{-1}), the second bit has a weight of 1/4: (2^{-2}), and so forth down to the LSB, which has a weight of 2^{-n}. The value of a binary number is obtained by adding up the weights of all non-zero bits. As an example, Table 1 lists the 16 permutations of 4-bits' worth of 1's and 0's, with their binary weights, and the equivalent numbers expressed as both decimal and binary fractions.

Decimal Fraction	Binary Fraction	Code: MSB (x1/2)	Bit 2 (x1/4)	Bit 3 (x1/8)	Bit 4 (x1/16)
0	0.0000	0	0	0	0
1/16 = 2^{-4} (LSB)	0.0001	0	0	0	1
2/16 = 1/8	0.0010	0	0	1	0
3/16 = 1/8 + 1/16	0.0011	0	0	1	1
4/16 = 1/4	0.0100	0	1	0	0
5/16 = 1/4 + 1/16	0.0101	0	1	0	1
6/16 = 1/4 + 1/8	0.0110	0	1	1	0
7/16 = 1/4 + 1/8 + 1/16	0.0111	0	1	1	1
8/16 = 1/2 (MSB)	0.1000	1	0	0	0
9/16 = 1/2 + 1/16	0.1001	1	0	0	1
10/16 = 1/2 + 1/8	0.1010	1	0	1	0
11/16 = 1/2 + 1/8 + 1/16	0.1011	1	0	1	1
12/16 = 1/2 + 1/4	0.1100	1	1	0	0
13/16 = 1/2 + 1/4 + 1/16	0.1101	1	1	0	1
14/16 = 1/2 + 1/4 + 1/8	0.1110	1	1	1	0
15/16 = 1/2 + 1/4 + 1/8 + 1/16	0.1111	1	1	1	1

Table 1. Fractional binary codes.

When all bits are "1" in natural binary, the number value is 1 - 2^{-n}, or normalized full-scale less 1 LSB (1 - 1/16 = 15/16 in the example). Strictly speaking, the number that is represented, written with an "integer point," is 0.1111 (= 1 - 0.0001). However, it is almost universal practice to write the code simply as the integer 1111 (i.e., "15") with the fractional nature of the corresponding number understood ["1111" → 1111/(1111 + 1), or 15/16].

For convenience, Table 2 lists bit weights in binary, for numbers having up to 20 bits. The practical range for the vast majority of applications is about 16 bits; for larger numbers of bits than 20, continue to divide by 2.

BIT	2^{-n}	$1/2^n$ (Fraction)	"dB"	$1/2^n$ (Decimal)	%	ppm
FS	2^0	1	0	1.0	100	1,000,000
MSB	2^{-1}	1/2	-6	0.5	50.	500,000
2	2^{-2}	1/4	-12	0.25	25	250,000
3	2^{-3}	1/8	-18.1	0.125	12.5	125,000
4	2^{-4}	1/16	-24.1	0.0625	6.2	62,500
5	2^{-5}	1/32	-30.1	0.03125	3.1	31,250
6	2^{-6}	1/64	-36.1	0.015625	1.6	15,625
7	2^{-7}	1/128	-42.1	0.007812	0.8	7,812
8	2^{-8}	1/256	-48.2	0.003906	0.4	3,906
9	2^{-9}	1/512	-54.2	0.001953	0.2	1,953
10	2^{-10}	1/1,024	-60.2	0.0009766	0.1	977
11	2^{-11}	1/2,048	-66.2	0.00048828	0.05	488
12	2^{-12}	1/4,096	-72.2	0.00024414	0.024	244
13	2^{-13}	1/8,192	-78.3	0.00012207	0.012	122
14	2^{-14}	1/16,384	-84.3	0.000061035	0.006	61
15	2^{-15}	1/32,768	-90.3	0.0000305176	0.003	31
16	2^{-16}	1/65,536	-96.3	0.0000152588	0.0015	15
17	2^{-17}	1/131,072	-102.3	0.00000762939	0.0008	7.6
18	2^{-18}	1/262,144	-108.4	0.000003814697	0.0004	3.8
19	2^{-19}	1/524,288	-114.4	0.000001907349	0.0002	1.9
20	2^{-20}	1/1,048,576	-120.4	0.0000009536743	0.0001	0.95

TABLE 2. Binary bit weights or resolution.

The weight assigned to the LSB is the *resolution* inherent in numbers having *n* bits. The "dB" column represents the logarithm (base 10) of the ratio of the LSB value to unity (full scale), multiplied by 20, in the popular manner. Each successive power of 2 represents a change of 6.02dB [i.e., 20 $\log_{10}$ (2)] or "6dB/octave."

In natural binary, the normalized numerical value of the code 1 0 1 1 1 0 1 0 1, a 9-bit code, would be

0.5000	MSB	1/2	=	256/512	
0.1250	Bit 3	1/8	=	64/512	
0.0625	Bit 4	1/16	=	32/512	
0.0312	Bit 5	1/32	=	16/512	
0.0078	Bit 7	1/128	=	4/512	
+0.0020	Bit 9 (LSB)	1/512	=	+ 1/512	
0.7285				373/512	= 0.7285

Bit numbering in some purely numerical devices, such as microprocessors, may be based on whole numbers, instead of binary fractions. In such systems, the LSB is always Bit 0 (viz., 2^0), the MSB is always Bit n - 1 (viz., 2^{n-1}), and the value of a binary number, normalized to full scale, is the integer value, divided by 2^n. An example of this approach can be found in Chapter I-4, in the section on interfacing converters with microprocessors.

BASIC CONVERSION RELATIONSHIPS

Perhaps the most fruitful way of indicating the relationship between analog and digital quantities involved in a conversion is to plot a graph. Since there are two complementary conversion relationships to be discussed, two graphs must be plotted, one for A/D conversion, the other for D/A conversion.

Figure 1 shows the graph for an ideal 3-bit D/A converter. A 3-bit converter has 8 dis-

crete coded levels, thus a total of 8 different inputs and 8 corresponding outputs, ranging from zero to 7/8 of "ful scale." Since no other levels can exist with this coding, it is plotted as a bar graph.

In practical D/A converters, the zero bar may not exactly zero (offset error). The range from zero to 7/8 F.S. may not be exactly as specified (gain error), the differences in heights of the bars may not be equal or changing uniformly (nonlinearity), and – in fact – if the nonlinearity is great enough, one or more values of analog output may be less than the values corresponding to codes having smaller weight (non-monotonic due to excessive *differential nonlinearity*). These errors (and others), the means of specifying and testing them, and some of the design techniques for keeping them small, are discussed in chapters II-2, II-3, II-4, and II-5.

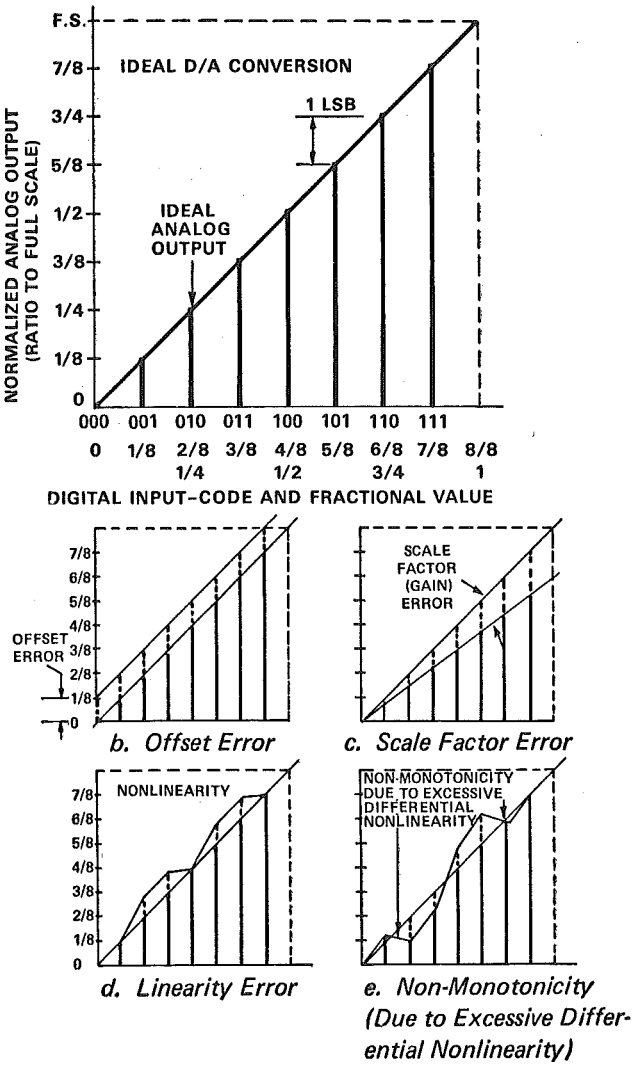


Figure 1. Conversion relationship in a 3-bit D/A converter
a. Ideal relationship
b, c, d, e. Typical sources of error

To visualize the ideal performance of converters having larger numbers of bits, one may intensify this pattern by interpolating additional bars between the bars on this graph. For example, a fourth bit would require 8 additional bars with heights halfway between the levels indicated. The value of the LSB would be F.S./16, and the maximum value would be 7/8 + 1/16 = 15/16 F.S. The next bit would interpolate 16 additional bars, the new LSB would be F.S./32, and the maximum value would be 31/32, etc. The straight line connecting the tops of the bars is the locus of the *envelope* of the ideal conversion relationship.

Figure 2 shows the graph for an ideal 3-bit A/D converter. Since all values of the analog input are presumed to exist, they must be *quantized* by partitioning the continuum into 8 discrete ranges. All analog values within a given range are represented by the same digital code, which corresponds to the nominal mid-range value. These mid-range values correspond to the bar heights of the D/A converter.

There is, therefore, in the A/D conversion process, an inherent *quantization uncertainty* of ±½ LSB, in addition to the conversion errors analogous to those existing for the D/A converter. The only sure way to reduce this quantization uncertainty is to increase the number of bits. (There are, of course, statistical interpolation tricks that may be performed in the digital processing or in analog filtering following subsequent D/A conversion, which will fill in missing analog values for large, rapidly-varying signals, but they will do nothing to indicate the variations within a quantum for an apparently-constant digital number.)

Since it is easier* to determine the location of a *transition* than it is to determine a mid-range value, errors and settings of A/D converters are defined and measured in terms of the analog values at which transitions occur, in relation to the ideal transition values. Like D/A converters, A/D converters have offset error: the first transition may not occur at exactly +1/2 LSB; scale-factor (or gain) error: the difference between the values at which the first transition and the last transition occur is not equal to (F.S. – 2LSB); and *linearity* error: the differences between transition values are not all equal or uniformly changing. If the *differential linearity* error is large enough, it is possible for one or more codes to be missed (the counterpart of non-monotonic D/A conversion).

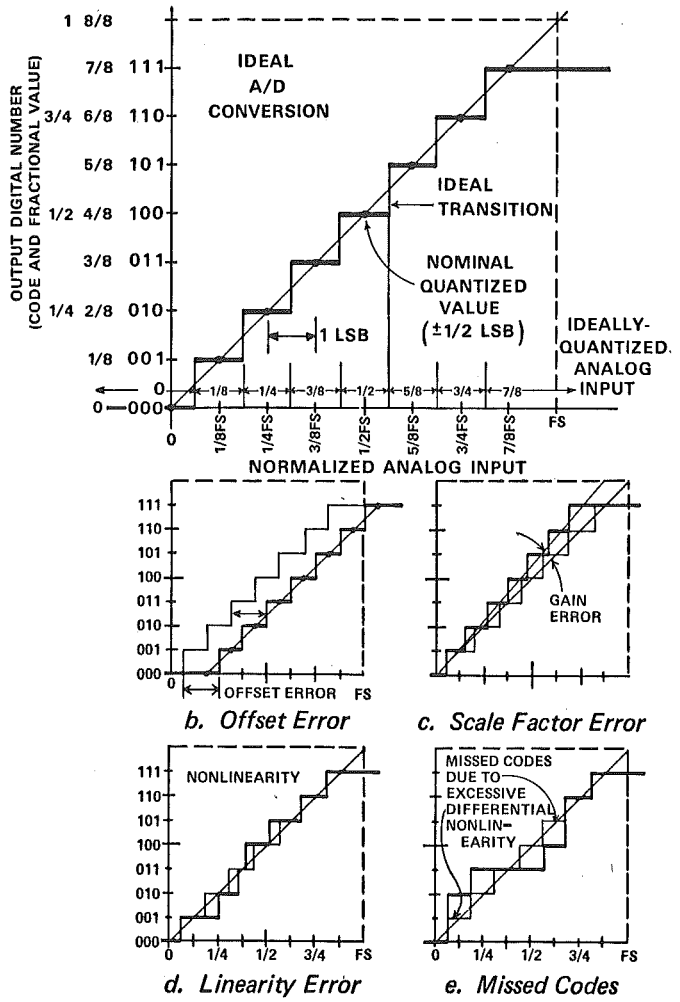


Figure 2. Conversion relationship in 3-Bit A/D converter
a. Ideal relationship b, c, d, e Typical sources of error.

*(using analog techniques)

An important factor in the conversion relationship is the choice of "Full Scale," the LSB magnitude, and the transition points. For a great many converters, full scale is in the vicinity of 10 volts: either exactly 10V or 10.24V. For 10V, the bit values are easily expressed as negative powers of 2, multiplied by 10; for 10.24V, the LSB can be expressed in "round" numbers, being a multiple or submultiple of 10mV.

Table 3 lists the LSB values, the "all 1's" value (i.e., F.S. – 1 LSB), and the A/D converter transition values at 1/2 LSB (for zero adjustment) and all 1's (F.S. – 1½ LSB, for scale factor adjustment) for resolutions to 2^{-20}, for both 10V and 10.24V full scale. If full scale is 5V (also a popular value), simply divide the appropriate numbers by 2.

No. of Bits	10V Full Scale				10.24V Full Scale			
			A/D Transitions				A/D Transitions	
			To	To			To	To
n	LSB	All 1's (Volts)	LSB (1/2 LSB)	All 1's (Volts)	LSB	All 1's (Volts)	LSB (+1/2 LSB)	All 1's (Volts)
1	5V	5.0	2.5V	2.5	5.12V	5.12	2.56V	2.56
2	2.5V	7.5	1.25V	6.25	2.56V	7.68	1.28V	6.40
3	1.25V	8.75	625mV	8.13	1.28V	8.96	640mV	8.32
4	625mV	9.38	312mV	9.07	640mV	9.60	320mV	9.28
5	312mV	9.69	156mV	9.53	320mV	9.92	160mV	9.76
6	156mV	9.84	78.1mV	9.76	160mV	10.08	80mV	10.00
7	78.1mV	9.92	39.1mV	9.88	80mV	10.16	40mV	10.12
8	39.1mV	9.961	19.5mV	9.941	40mV	10.20	20mV	10.18
9	19.5mV	9.980	9.77mV	9.970	20mV	10.220	10mV	10.21
10	9.77mV	9.990	4.88mV	9.985	10mV	10.230	5mV	10.225
11	4.88mV	9.9951	2.44mV	9.9927	5mV	10.235	2.5mV	10.232
12	2.44mV	9.9976	1.22mV	9.9964	2.5mV	10.2375	1.25mV	10.2362
13	1.22mV	9.9988	610μV	9.9982	1.25mV	10.2388	625μV	10.2382
14	610μV	9.9994	305μV	9.9991	625μV	10.2394	312μV	10.2391
15	305μV	9.99970	153μV	9.99955	312μV	10.23969	156μV	10.23953
16	153μV	9.99985	76μV	9.99977	156μV	10.23984	78.1μV	10.23976
17	76μV	9.99992	38μV	9.99988	78.1μV	10.23992	39.1μV	10.23988
18	38μV	9.999962	19μV	9.999943	39.1μV	10.239961	19.5μV	10.239941
19	19μV	9.999981	9.5μV	9.999971	19.5μV	10.239980	9.77μV	10.239970
20	9.5μV	9.999990	4.8μV	9.999985	9.77μV	10.239990	4.88μV	10.239985

Table 3. LSB and (FS - LSB) values for 10V and 10.24V conversion.

OTHER CODES

Although binary is the most commonly-used code, there are a number of other popular codes used at system interfaces, depending on signal range and polarity, conversion technique, specially desired characteristics, and origin or destination of digital information.

BINARY-CODED DECIMAL (BCD)

This is a code in which each decimal digit is represented by a group of 4 binary-coded digits. The LSB of the most significant group, or "quad," had a weight of 0.1, the LSB of the next has a weight of 0.01, the LSB of the next has a weight of 0.001, etc. Each quad has 10 permissible levels with weights 0 to 9. Group values in excess of 9 are not permitted. For example, Table 4 shows BCD coding for a variety of numbers between 0 and 0.99.

A/D converters with the BCD code are used primarily in digital voltmeters and panel meters, since each quad's output may be decoded to drive a numeric display using the familiar decimal numbers. If the display is of a BCD digitally transmitted or processed number, or if the input is via a thumbwheel switch, a D/A converter that responds to BCD may be used to furnish an analog output from the same digital input.

BCD is somewhat wasteful, in the sense that each BCD quad has 10/16 the resolution of a comparable natural binary quad. Table 5 shows the relative resolution capability.

Decimal Fraction	BCD Code MSQ (x1/10) x8 x4 x2 x1	2nd Quad (x1/100) x8 x4 x2 x1
0.00 = 0.00 + 0.00	0 0 0 0	0 0 0 0
0.01 = 0.00 + 0.01	0 0 0 0	0 0 0 1
0.02 = 0.00 + 0.02	0 0 0 0	0 0 1 0
0.03 = 0.00 + 0.03	0 0 0 0	0 0 1 1
0.04 = 0.00 + 0.04	0 0 0 0	0 1 0 0
0.05 = 0.00 + 0.05	0 0 0 0	0 1 0 1
0.06 = 0.00 + 0.06	0 0 0 0	0 1 1 0
0.07 = 0.00 + 0.07	0 0 0 0	0 1 1 1
0.08 = 0.00 + 0.08	0 0 0 0	1 0 0 0
0.09 = 0.00 + 0.09	0 0 0 0	1 0 0 1
0.10 = 0.10 + 0.00	0 0 0 1	0 0 0 0
0.11 = 0.10 + 0.01	0 0 0 1	0 0 0 1
:		
0.20 = 0.20 + 0.00	0 0 1 0	0 0 0 0
:		
0.30 = 0.30 + 0.00	0 0 1 1	0 0 0 0
:		
0.90 = 0.90 + 0.00	1 0 0 1	0 0 0 0
0.91 = 0.90 + 0.01	1 0 0 1	0 0 0 1
:		
0.98 = 0.90 + 0.08	1 0 0 1	1 0 0 0
0.99 = 0.90 + 0.09	1 0 0 1	1 0 0 1

Table 4. Examples of 2-digit BCD weighting.

Number of Bits	Least Significant Bit Binary	BCD	Number of Binary Bits Needed For Same Resolution as BCD
4	0.062	0.1	4
8	0.0039	0.01	7
12	0.00024	0.001	10
16	0.000015	0.0001	14
20	0.000001	0.00001	17
24	0.0000002	0.000001	20

Table 5. Relative resolution of BCD and binary.

OVERRANGING

Many BCD A/D converters have an additional bit with weight equal to full scale, in a position "more significant" than the MSB.

This additional bit provides a maximum of 100% "overrange" capability. Additional "super-significant" bits would provide binary 300% (2 bits) and 700% (3 bits) overrange capability (or extend the range to nearly 800% of the BCD "full scale.") The overrange bit is most commonly used in digital voltmeters and panel meters to indicate that nominal full scale has been exceeded and that the visual reading may be erroneous.

Overrange bits need not be restricted to BCD. They are useful as "flags" in any conversion process for which an overrange input would given an ambiguous reading, or where an overrange input indicates anomalous analog system behavior. The overrange bit must of course be of suitable accuracy, since it is, in effect, the MSB.

2-4-2-1 BINARY-CODED DECIMAL

This is a code that is still in use, in which the bit in the MSB position in each quad has a weight of 2 instead of the usual 8 that is normal for BCD. It is found, for example, at

the digital output of some older Hewlett-Packard digital voltmeters. The relative weights within a quad are given in Table 6.

		2☆ (x2)	4 (x4)	2 (x2)	1 (x1)
0.0		0	0	0	0
0.1 =	0.1	0	0	0	1
0.2 =	0.2	0	0	1	0
0.3 =	0.1 + 0.2	0	0	1	1
0.4 =	0.4	0	1	0	0
0.5 =	0.1 + 0.4	0	1	0	1
0.6 =	0.2 + 0.4	0	1	1	0
0.7 =	0.1 + 0.2 + 0.4	0	1	1	1
0.8 =	0.2 + 0.4 + 0.2	1	1	1	0
0.9 =	0.1 + 0.2 + 0.4 + 0.2	1	1	1	1

Table 6. 2-4-2-1 BCD bit weights within each quad.

This code was more economical to implement in the days before integrated-circuit logic became common, still has the advantages of having all 1's for (full scale – LSB) and requiring a smaller range of resistance in D/A-converter ladder networks based on binary conductance values.

GRAY CODE

This is a binary code in which the bit position does not signify a numerical weighting; however, in converters using it, each code still corresponds to a unique portion of the analog range. It is easily translatable into natural binary (Table7):

Decimal Fraction	Gray Code				Binary Code			
0	0	0	0	0	0	0	0	0
1/16	0	0	0	1	0	0	0	1
2/16	0	0	1	1	0	0	1	0
3/16	0	0	1	0	0	0	1	1
4/16	0	1	1	0	0	1	0	0
5/16	0	1	1	1	0	1	0	1
6/16	0	1	0	1	0	1	1	0
7/16	0	1	0	0	0	1	1	1
8/16	1	1	0	0	1	0	0	0
9/16	1	1	0	1	1	0	0	1
10/16	1	1	1	1	1	0	1	0
11/16	1	1	1	0	1	0	1	1
12/16	1	0	1	0	1	1	0	0
13/16	1	0	1	1	1	1	0	1
14/16	1	0	0	1	1	1	1	0
15/16	1	0	0	0	1	1	1	1

Table 7. Comparison of 4-Bit binary and Gray codes. Underlined bits indicate changes as number increases.

In Gray code, as the number value changes, the transitions from one code to the next involve only one bit at a time. The bits that change as the numbers increase are underlined in the table.

The conversion from binary to Gray code occurs as follows: If the binary MSB is zero, the Gray code MSB will be zero. Then, continuing to read from MSB to LSB, each change produces a "1," each non-change produces a "0." For example, .11 in binary, 1011, becomes 1110 in Gray code (1 → 1, 1-to-0 → 1, 0-to-1 → 1, 1-to-1 → 0). Another example: the 12-bit binary number 101111000101 becomes 111000100111. Fig. 3 shows one way in which binary to Gray code conversion may be mechanized.

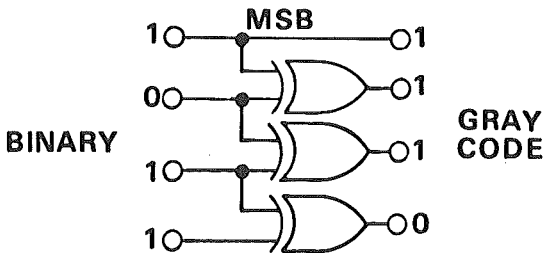


Figure 3. Binary to Gray code conversion using exclusive – or gates.

The conversion from Gray code to binary is just the reverse of the conversion from binary to Gray code: The binary MSB will be the same as the Gray code MSB. Then, continuing to read from MSB to LSB, if the next bit is 1, the next binary bit is the complement of the previous binary bit. For example, if the Gray code is 1110, the corresponding binary is 1011 (1 → 1, 1 → 1-to-0, 1 → 0-to-1, 0 → 1-to-1). Another example, the 8-bit Gray code 01110000 is 01011111 in binary. A mechanization of Gray code-to-binary conversion appears in Figure 4.

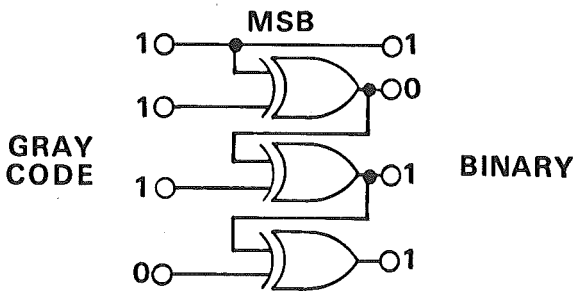


Figure 4. Gray code to binary conversion.

Gray code is useful for shaft encoders (angle-to-digital converters) because the change of only 1 bit for each increment eliminates false intermediate codes that could occur in natural binary conversion. Here, for comparison, are Gray code and binary developed optical shaft encoders for 4-bit resolution.

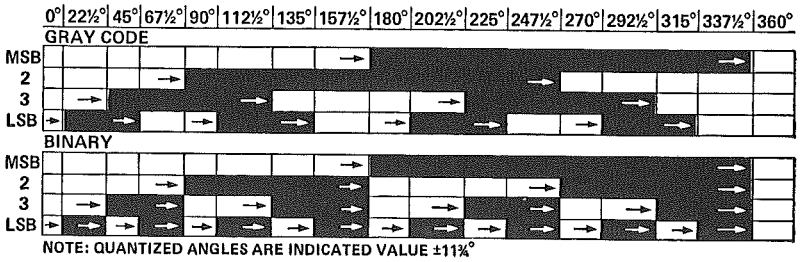


Figure 5. Gray vs. binary encoding.

Note that, with the Gray code converter, there is only one bit-change at each transition. If the edge of a shaded area is slightly out of line, the coding will be in error by a small fraction of an LSB. In the binary converter, *all four bits change at once* at the 180° and and 360° transitions. If bit 2's shaded area were to end a little to the left of the 180° transition, the code, in a small region, would be 0011, indicating the 67½° range, or a fictitious progression from 157½° to 67½°to 180°. We leave the catastrophic implications of this to the reader.

The shaft encoder is a *simultaneous* converter: all bits appear at once and can be read in parallel at any time. There is an electrical equivalent form of simultaneous A/D converter having a Gray code output. It employs a chain of biased comparators, the outputs of which provide a quantized indication of the analog input level: all comparators above it

are 0, all comparators below it are 1. Multi-input logic gates then make the decisions necessary to obtain a parallel Gray code output. Such converters are quite fast, some being capable of producing 10M meaningful conversions per second, but they require a number of comparisons that is a geometric function of the required resolution, (i.e., $2^n - 1$), as well as logic gates having large numbers of inputs.

A variation of this scheme, the cyclical converter, which also has Gray code output, uses fewer comparators, with more-accurate output states, but it requires more time to perform the conversion. It continuously tracks the analog input.

The use of Gray code in fast converters that provide continuous conversions has the same rationale as for the shaft encoder. Any Gray code output value that is latched into a register will always be within ±1 LSB of the correct value, even if the latching occurs just as a bit is switching. With binary, however, where many bits can switch at a single transition, it is possible to latch in mid-flight, and, because of the "skew" between turn-on and turn-off speeds, lock in an utterly false code.

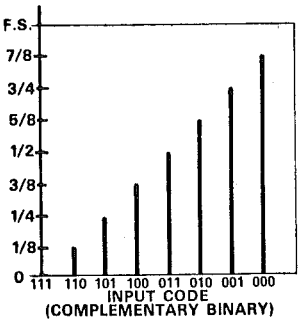
COMPLEMENTARY CODES

The actual mechanization of some forms of converters, (for example, D/A converters using monolithic quad current switches) may require codes, such as natural binary or BCD, in which all bits are represented by their complements. Such codes are called complementary codes.

In a 4-bit complementary-binary converter, 0 is represented by 1111, half-scale (MSB) by 0111, and full scale (less 1LSB) by 0000. It can be easily obtained from the "$\bar{Q}$" outputs of a register, of which "Q" is the normal output sense.

Similarly, for each quad of a BCD-coded converter, *complementary* BCD is the code obtained by representing all bits by their complements. In complementary BCD, 0 is represented by 1111, and 9 is represented by 0110. As an example, Table 8 lists the equivalents for 1 through 11 in complementary binary and complementary BCD (with overrange bit).

Decimal Number	Natural Binary	Complementary Binary	BCD	Complementary BCD
0 BIN DEC	0000	1111	00000	11111
1, 1/16, 1/10	0001	1110	00001	11110
2, 2/16, 2/10	0010	1101	00010	11101
3, 3/16, 3/10	0011	1100	00011	11100
4, 4/16, 4/10	0100	1011	00100	11011
5, 5/16, 5/10	0101	1010	00101	11010
6, 6/16, 6/10	0110	1001	00110	11001
7, 7/16, 7/10	0111	1000	00111	11000
8, 8/16, 8/10	1000	0111	01000	10111
9, 9/16, 9/10	1001	0110	01001	10110
10,10/16,10/10	1010	0101	10000	01111
11,11/16,11/10	1011	0100	10001	01110



Conversion Relationship of Ideal 3-Bit Complementary Binary D/A Converter

Table 8. Complementary codes.

If a natural binary input were applied to a D/A converter coded to respond to complementary binary, the output would be in reverse order, i.e., zero output for all 1's, and F.S. – 1 LSB for all 0's.

The complementary codes discussed above involve complementing *all bits,* for convenience in implementing the conversion relationship using certain kinds of switches (i.e., those that respond to complementary logic). We could just as well have left the logic unchanged but redefined it as "negative true." However, for consistency in elucidation, we define all logic in terms of "positive true" TTL (or DTL), as explained at the beginning of the

chapter. It is important to understand that, for purposes of this discussion, these complementary codes have nothing to do with representation of the *analog polarity* relationship, a matter that will be discussed next.

ANALOG POLARITY

So far, the conversion relationships mentioned have been unipolar: the codes have represented numbers, which in turn represent the normalized *magnitudes* of analog variables,[2] without regard to polarity. A unipolar A/D converter will respond to analog signals of only one polarity, and a unipolar D/A converter will produce analog signals of only one polarity.

The analog signal polarity is determined either by using a converter whose reference and switches (or specifications) are compatible with the desired analog polarity, or (if for reasons of economy or availability, a converter is available having predetermined polarity) by operating on the analog signal before A/D conversion – or after D/A conversion – to invert its polarity, and also perform any necessary scale changes, if range must be adapted too.

BIPOLAR CODES

For conversion of bipolar analog signals into a digital code that retains sign information, one extra bit – the "sign bit" – is necessary. This "most significant bit" doubles the analog range and halves the peak-to-peak resolution. In some cases, the sign bit is provided by re-interpreting the existing MSB, in which event the analog *range* may still be doubled, but the *resolution* is twice as coarse. For example, if a 10-bit converter's resolution is 1/1024, for the range 0-10V, we may use a bipolar code having 11 bits, with peak-to-peak resolution of 1/2048 and range of ±10V, or retain a code having 10 bits, but "stretch" the range to ±10V, in which case the peak-to-peak resolution remains 1/1024, which doubles the magnitude of the LSB. It must be emphasized that, because the sign digit doubles both the range *and* the number of levels, the LSB's ratio to full scale in either polarity is $2^{-(n-1)}$, not 2^{-n}.

The most-often-used binary codes in bipolar conversion are: Sign-magnitude (magnitude plus sign), Offset binary, Two's complement, and One's complement. Table 9 shows each of these codes expressed for 4 bits (3 bits plus sign).

	Decimal Fraction					
Number	Positive Reference	Negative Reference	Sign + Magnitude	Two's Complement	Offset Binary	One's Complement
+7	+7/8	−7/8	0 1 1 1	0 1 1 1	1 1 1 1	0 1 1 1
+6	+6/8	−6/8	0 1 1 0	0 1 1 0	1 1 1 0	0 1 1 0
+5	+5/8	−5/8	0 1 0 1	0 1 0 1	1 1 0 1	0 1 0 1
+4	+4/8	−4/8	0 1 0 0	0 1 0 0	1 1 0 0	0 1 0 0
+3	+3/8	−3/8	0 0 1 1	0 0 1 1	1 0 1 1	0 0 1 1
+2	+2/8	−2/8	0 0 1 0	0 0 1 0	1 0 1 0	0 0 1 0
+1	+1/8	−1/8	0 0 0 1	0 0 0 1	1 0 0 1	0 0 0 1
0	0+	0−	0 0 0 0	0 0 0 0	1 0 0 0	0 0 0 0
0	0−	0+	1 0 0 0	(0 0 0 0)	(1 0 0 0)	1 1 1 1
−1	−1/8	+1/8	1 0 0 1	1 1 1 1	0 1 1 1	1 1 1 0
−2	−2/8	+2/8	1 0 1 0	1 1 1 0	0 1 1 0	1 1 0 1
−3	−3/8	+3/8	1 0 1 1	1 1 0 1	0 1 0 1	1 1 0 0
−4	−4/8	+4/8	1 1 0 0	1 1 0 0	0 1 0 0	1 0 1 1
−5	−5/8	+5/8	1 1 0 1	1 0 1 1	0 0 1 1	1 0 1 0
−6	−6/8	+6/8	1 1 1 0	1 0 1 0	0 0 1 0	1 0 0 1
−7	−7/8	+7/8	1 1 1 1	1 0 0 1	0 0 0 1	1 0 0 0
−8	−8/8	+8/8		(1 0 0 0)	(0 0 0 0)	

Table 9. Commonly-used bipolar codes.

[2]Gray code is an exception. Since it is not quantitatively weighted, it can represent any arbitrary range of magnitudes of any polarity.

Because the analog signal now has a choice of polarity, we must be careful about the relationship between the code and the polarity of the analog signal. "Positive reference" indicates that the analog signal[3] increases positively as the digital number increases. "Negative reference," on the other hand, indicates that the analog signal decreases towards negative full scale as the digital number increases. Conversion relationships for bipolar A/D and D/A converters are shown graphically in Figures 6 and 7.

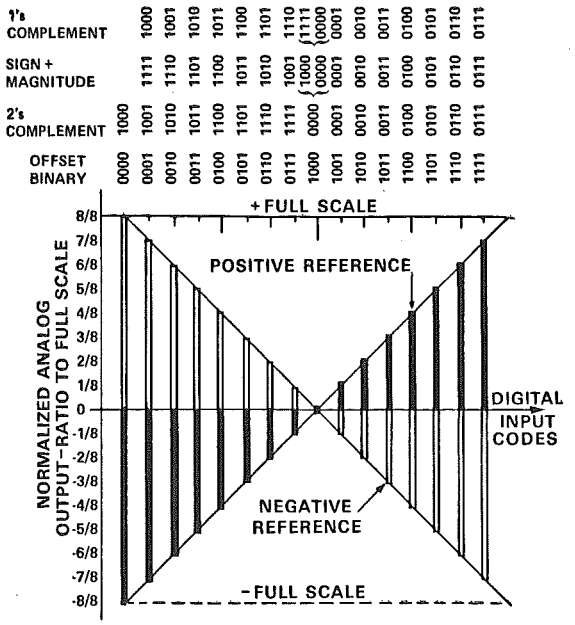


Figure 6. Ideal D/A conversion realtionship for 4-bit (3-bit-plus-sign) offset binary, 2's complement, sign-magnitude, 1's complement codes.

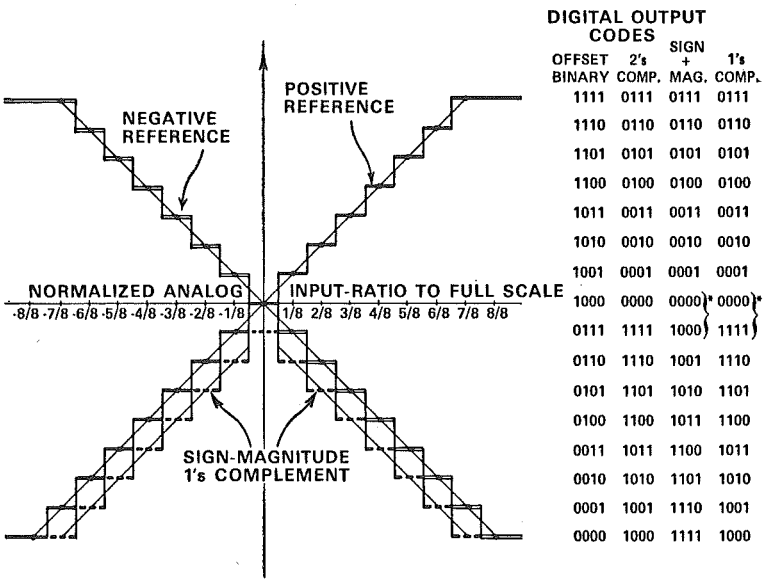


Figure 7. Ideal A/D conversion relationship for 4-bit (3-bit-plus-sign) offset binary, 2's complement, sign magnitude, 1's complement codes.

Sign-Magnitude would appear to be the most straightforward way of expressing signed analog quantities digitally. Simply determine the code appropriate for the magnitude and add a polarity bit. It is used advantageously in D/A converters that operate in the vi-

[3] A/D converter input or D/A converter output.

*In sign-magnitude and 1's-complement A/D converters, the ambiguous zero must be dealt with. In some units, one of the codes is "forbidden," (DVM's generally read "+0" only), in others, the ±½ LSB zero region is divided into two regions (0 to +½ LSB, and 0 to -½ LSB), one of which produces one code, the other the other code. This is a more useful approach in codes having a sequential continuum of numbers, such a 1's complement.

cinity of zero, where the application calls for smooth and linear transitions from positive small voltages to negative small voltages. As can be seen in the example in the table, it is the only code for which the three magnitude bits do not have a major transition (all 1's to all 0's, or equivalent) at zero. Sign-magnitude BCD is almost universally used for bipolar digital voltmeters (A/D converters).

It does have some shortcomings, though. In data-processing applications, the *other* codes are more-readily usable for computation with a minimum of translation. One of its problems is that it has two codes for zero. For this reason, sign-magnitude is harder to interface with digitally, because it requires either software, or additional equipment. In addition, the converter circuitry for sign-magnitude is usually somewhat more complicated and costly than for some other codes.

Offset binary is the easiest code to embody with converter circuitry. An examination of the offset binary code for three bits plus sign will show that it is really a natural binary code for four bits, except that its zero is at negative full scale, the LSB is 1/16 of the bipolar range, and the MSB is turned on at analog zero. Therefore, to make an offset binary 3-bits-plus sign converter out of a 4-bit D/A converter having 0-to-10V full-scale range, we have only to double its scale factor (20V range), and offset its zero by one half of the full range (-10V), an operation which is neither difficult nor expensive. Similarly, for an A/D converter, one would attenuate the input by one-half, and add a bias of one-half the full range.

Besides its ease of implementation, offset binary has the further advantage that it is compatible with computer inputs and outputs, it is easily changed to the more-computationally useful two's complement (just complement the MSB), and it has a single unambiguous code for zero. The all-zeros negative full scale code (0000), though not used in computing (because – (F.S. - 1 LSB) is the most negative value defined in computing), is nevertheless useful as a converter checking and adjustment code.

The principal drawback of offset binary is that a major bit transition occurs at 0 (all bits change, from 0111 to 1000). This can lead to "glitch" problems dynamically (the difference in speed between bits turning on and off can lead to large spikes) and to linearity problems statically (the largest linearity errors are most likely to occur at major transitions, because the transition is essentially a difference between two large numbers). Also, in offset binary, zero errors may be greater than with sign-magnitude, because the zero analog level is usually obtained by taking a difference between the MSB (½ full range) and a bias (½ full range), again, two large numbers.

Two's complement, for conversion purposes, consists of a binary code for positive magnitudes (0 sign bit), and the two's complement of each positive number to represent its negative. The two's complement is formed arithmetically by complementing the number and adding 1 LSB. For example, the two's complement of 3/8 (0011) would be its complement plus 1 LSB, or 1100 + 0001 = 1101.

Two's complement is a useful code computationally because it can be thought of as a set of negative numbers. Therefore, addition can be used instead of subtraction. For example, to subtract 3/8 from 4/8, add 4/8 to -3/8, or 0100 to 1101. The result is 0001 (disregarding the extra carry), or 1/8.

If the two's complement code and the offset binary code are compared, it can be seen that the only difference between them is that the MSB of one is replaced by its complement in the other (Nature's way of helping converter manufacturers and users). Since both a digit and its complement are available from most flip-flops, an offset-binary-coded converter may be used for 2's complement, just by using the MSB's complement at the output of an A/D converter or at the output of a D/A converter's input register. And vice versa.

Two's complement has the same disadvantages as offset binary, since the conversion process is identical.

One's complement is a means of representing negative numbers sometimes used by computers. One's complement is obtained arithmetically by simply complementing all of a number's digits. Thus, the one's complement of 3/8 (0011) is (1100). When a number is subtracted by adding its 1's complement, the extra carry (that is disregarded in 2's complement), if present, cause 1 LSB to be added to the total ("end-around carry.") Thus, subtracting 3/8 from 4/8, 0100 + 1100 = 0000 + 0001 = 0001 (or 1/8). A one's complement code can be formed by complementing each positive value to obtain its corresponding negative value, including – alas – zero, which is then represented by two codes, 0000 and 1111.

Besides its ambiguous zero, another disadvantage of this code in conversion is that it is not as readily implemented as is 2's complement. If it is not converted to 2's complement before a D/A conversion, by adding 1 LSB digitally when the MSB is 1 (indicating a negative number), then the easiest way to implement the conversion is by performing a 2's complement conversion, and adding the *analog* value of 1LSB, if the MSB is 1. Adding the extra analog bit can be done simply and elegantly (if not accurately) by resistively dividing the digital MSB logic level down to the LSB's analog value and summing this attenuated signal.

CODE CONVERSION

Since code conversion may be desirable, either after A/D conversion or before D/A conversion, in order to make it possible to use a converter that produces the best results at the lowest cost, the matrix of Table 10 briefly outlines the relationships among the codes.

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

Table 10. Relations among bipolar codes.

OTHER BIPOLAR CODES

The list of bipolar codes mentioned above may seem quite complete and coherent from the tutorial point of view, but it does not fully reflect the ingenuity and diversity of the computer and converter industries. There are a number of variations in more-or-less widespread usage that should be mentioned here because they will inevitably be encountered. Fortunately, they are based on codes we have already discussed and may be easily described.

Modified sign-magnitude: This is a version of sign-magnitude in which the MSB is complemented (1 for positive, 0 for negative).

Modified one's complement: Like modified sign-magnitude, a version in which the MSB is complemented (1 for positive, 0 for negative).

Both of the above codes have polarity bits that are the same as for offset binary – which is the lone standout in Table 9. Since offset binary is popular among converter manufacturers, it stands to reason that other codings should be available with compatible sign bits, for the sake of uniformity, even though they may take us one step away from the basic natural binary rationale.

Complementary everything: All of the above-mentioned codes may be completely complemented to form complementary sign-magnitude, complementary offset binary, complementary 2's complement, and complementary 1's complement. (These are, as explained earlier in this chapter, "negative true" versions.) Such codes, although they made life a little more complex, were necessary to take advantage of some of the best monolithic switching hardware available at one time – quad current switches. Fortunately, A/D converters and D/A converters that are supplied with registers performed the complementarity internally, so that the user wasn't usually aware of the complexities that lay within.[4] However, users of monolithic and hybrid converters without registers should be prepared to adjust their thinking (and especially their test equipment) to include the possible application of complementary codes.

For the sake of completeness, Table 11 lists the codes mentioned above, for 3-bits-plus-sign.

			COMPLEMENTARY CODES			
	Modified Sign-	Modified 1's	Comp. Sign-	Comp. Offset	Comp. 2's	Comp. 1's
Number	Magnitude	Complement	Magnitude	Binary	Complement	Complement
+7	1 1 1 1	1 1 1 1	1 0 0 0	0 0 0 0	1 0 0 0	1 0 0 0
+6	1 1 1 0	1 1 1 0	1 0 0 1	0 0 0 1	1 0 0 1	1 0 0 1
+5	1 1 0 1	1 1 0 1	1 0 1 0	0 0 1 0	1 0 1 0	1 0 1 0
+4	1 1 0 0	1 1 0 0	1 0 1 1	0 0 1 1	1 0 1 1	1 0 1 1
+3	1 0 1 1	1 0 1 1	1 1 0 0	0 1 0 0	1 1 0 0	1 1 0 0
+2	1 0 1 0	1 0 1 0	1 1 0 1	0 1 0 1	1 1 0 1	1 1 0 1
+1	1 0 0 1	1 0 0 1	1 1 1 0	0 1 1 0	1 1 1 0	1 1 1 0
0+	1 0 0 0	1 0 0 0	1 1 1 1	0 1 1 1	1 1 1 1	1 1 1 1
0-	0 0 0 0	0 1 1 1	0 1 1 1	0 1 1 1	1 1 1 1	0 0 0 0
-1	0 0 0 1	0 1 1 0	0 1 1 0	1 0 0 0	0 0 0 0	0 0 0 1
-2	0 0 1 0	0 1 0 1	0 1 0 1	1 0 0 1	0 0 0 1	0 0 1 0
-3	0 0 1 1	0 1 0 0	0 1 0 0	1 0 1 0	0 0 1 0	0 0 1 1
-4	0 1 0 0	0 0 1 1	0 0 1 1	1 0 1 1	0 0 1 1	0 1 0 0
-5	0 1 0 1	0 0 1 0	0 0 1 0	1 1 0 0	0 1 0 0	0 1 0 1
-6	0 1 1 0	0 0 0 1	0 0 0 1	1 1 0 1	0 1 0 1	0 1 1 0
-7	0 1 1 1	0 0 0 0	0 0 0 0	1 1 1 0	0 1 1 0	0 1 1 1
-8				1 1 1 1	0 1 1 1	

Table 11. Modified and complementary bipolar codes.

ARBITRARY BIASING AND SCALING

The conversion relationships discussed so far have been either strictly one-sided (0 to full scale) or symmetrical (±full scale). The reason for this emphasis is that most commercially-available converters are built that way – as general-purpose devices.

However, since the principal relationship between the analog variable and the digital number is *proportionality*, the repertoire of codes corresponding to a given resolution may represent any portion of the analog voltage or current range. For example, if one wished to encode the voltage range from 4 to 7 volts in binary, using a 0–10V A/D converter, one could simply apply the voltage without any transformation, using only 0.3 of the available number of bits. However, a more efficient alternative would be to offset the input by 4 volts, amplify by 3.33, and apply the resulting 0–10V signal to the converter,

[4] Since both polarities of logic are usually available, complementary inputs (or outputs) can *simplify* loading, and wiring, and fanout problems. For example, if an A/D converter uses a conversion process involving a complementary-input D/A converter, the complementary logic outputs may be applied to the D/A converter, and the uncomplemented outputs to the outside world.

thereby making use of the entire range of available codes and improving resolution by a greater than 3. In a sense, the conversion relationship between the original input and the digital output is an *offset binary* code. The subsequent digital processing would take this transformation into account via the software.

Another sort of arbitrary scaling might result if the analog signal were proportional to a temperature range of (for example) 0° to 70°C, and one desired a direct readout of temperature on a digital voltmeter. A typical approach might be to scale the voltage directly to the temperature numbers (e.g., 10°/V) and apply it to a DVM, with the location of the decimal point re-interpreted. The DVM would then provide a readout in *engineering units.*

In offset binary and 2's complement coding, a converter's output is asymmetrical, because a code exists for -F.S. (i.e., -1), but not for +F.S. ($1 - 2^{-(n-1)}$ is maximum). Since a computer will not use the code for -F.S., the remaining codes are symmetrical for computing purposes, and the -F.S. code is unused, and – in effect – lost. For some purposes, such as data transmissions, it may be desirable to retain the information in *all* codes, and achieve full scale and symmetry as well. This can be accomplished by biasing the analog signal by F.S./(2^n - 1), and multiplying the reference by $1/(1 - 2^{-n})$.[5] The only drawback of this scheme is that a D/A converter output, though providing a complete, symmetrical full-scale swing, will not have an analog zero state. (Fig. 8).

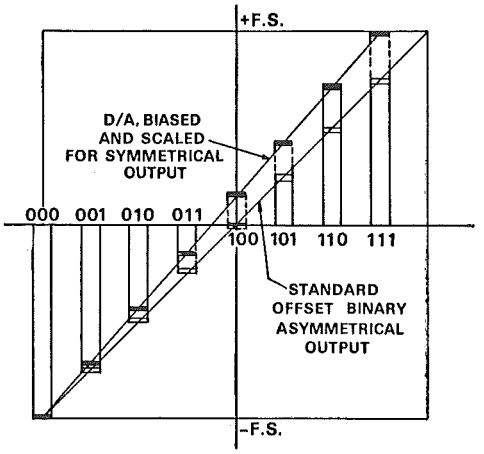


Figure 8. Conversion relationship for a 2-bit-plus-sign D/A converter biased and scaled for symmetrical analog output. Note that there is no code that gives analog zero.

DAC's AS MULTIPLIERS AND ADC's AS DIVIDERS

The D/A converter can be thought of as a digitally-controlled potentiometer that produces an analog output (voltage or current) that is a normalized fraction of its "full scale" setting. The output voltage or current depends on the reference value chosen to determine "full scale" output. If the reference may vary in response to an analog signal, the output is proportional to the product of the digital number and the analog input. The product's polarity depends on the analog signal polarity, and the digital coding and conversion relationship. 4-quadrant multiplication is available if the D/A converter accepts reference signals of both positive and negative polarities, and the digital response is bipolar. A typical conversion relationship for a 4-quadrant multiplying DAC having 3-bit-plus-sign 2's complement coding is shown in Figure 9, interpreting the Multiplying DAC as a digitally-controlled variable-gain amplifier.

In another interpretation, the envelope of the ideal bipolar D/A converter output in Figure 6 could be seen as proportional to the analog signal input, starting from full scale "Positive Reference," being attenuated as the analog signal is reduced, passing through zero, and increasing negatively to the "Negative Reference" envelope.

[5]These are b and m, respectively, in the equation $y = mx + b$.

Multiplying D/A converters may be 4-quadrant, two-quadrant (single polarity of either analog or digital variable), or one quadrant. They may even be fractional-quadrant, if the reference has a limited range of variation.

In analog-to-digital converters, the digital output number depends on the ratio of the quantized input to the "full-scale" reference. If the reference is allowed to change in response to a second analog input, the digital output will be proportional to the ratio of the analog signal to the reference signal. Thus, the "ratiometric" A/D converter can be thought of as an analog divider with digital output.

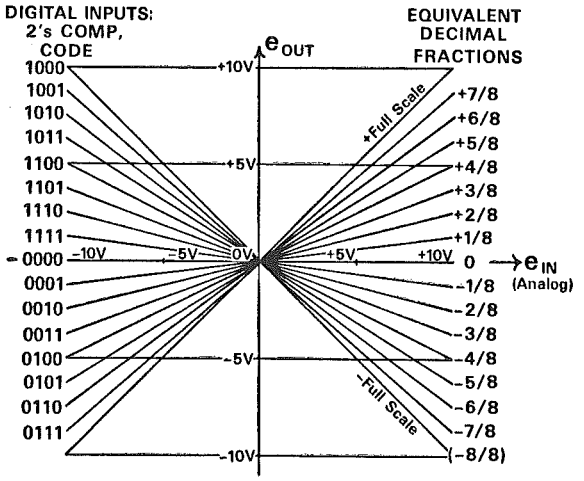


Figure 9. D/A converter as four-quadrant multiplier of analog voltage and 3-bit-plus-sign 2's complement digital number. Analog output vs. analog input as a function of digital input code.

ELECTRICAL INTERFACES WITH CONVERTERS

Converters may have associated with them six families of electrical inputs and outputs: analog signal, digital code, power, control, configuration, and reference. Table 12 indicates some of the properties of these interfaces, and the text that follows adds further detail.

	D/A Converters	A/D Converters
ANALOG SIGNAL	Output: Voltage or Current Polarity Magnitude	Input: Usually Voltage Polarity Magnitude
DIGITAL CODE	Input: Buffered or Direct Coding Logic Levels Format: Serial Parallel Byte-Serial	Output: Coding Logic Levels Timing (Clock) Format: Serial Parallel Byte-Serial
CONTROL	Input: Strobe(s)	Inputs: Convert Command Enables Clock Outputs: Status Overrange
CONFIGURATION	Serial/Parallel Short Cycle ( ) Bits Address	Byte-Enable Short Cycle ( ) Bits Address
POWER	Analog: Usually ±15V or same as Digital Digital: +5V (TTL) +5V to +15V (CMOS)	Analog: Usually ±15V or same as Digital Digital: +5V (TTL) +5V to +15V(CMOS) -5V to -15V (CMOS V_{SS})
REFERENCE	Internal or External Fixed or Variable Polarity	Internal or External Fixed or Varible Polarity

Table 12. Converter interfaces.

Figure 10 is a block diagram showing typical connections to a parallel converter. There may be yet other connections, such as a clock synchronization input or output, complementary logic inputs or outputs, and connections that are essentially internal but are brought out for the sake of optional flexibility, such as bipolar offset reference terminals.

Block diagrams used in this book (and much of the literature), for facility of communication, tend to depict only those interfaces that are of specific relevance to the point under discussion. However, the reader should be aware that "out of sight" should not mean "out of mind."

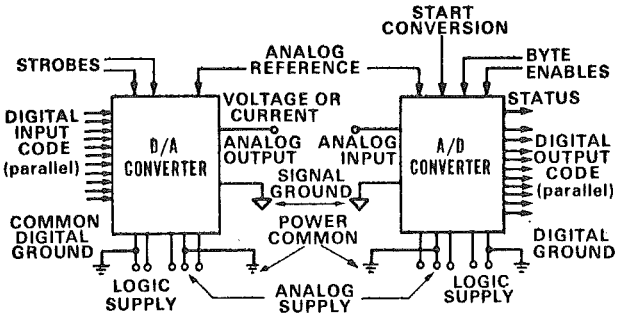


Figure 10. Converter connections.

GROUND RULE

The experienced circuit designer will recognize the feeling of wariness provoked by the presence of two supplies, and several classes of signals, all needing return "to ground." Grounding is indeed important to system performance; discussions of the essentials of grounding practice will be found (as appropriate) in several places in this book; however, for clarity in the present exposition, we will consider that all grounds are always at true zero potential with respect to all input and output signals. Accordingly, the discussions that follow will everywhere employ the inverted triangle, which represents ideal signal ground.

POWER SUPPLIES

The choice of power supplies for use with converters is governed by their effect on conversion accuracy, system noise, size and weight, reliability, and cost. Supply capacity is determined by the choice of system philosophy: one main supply feeding all elements, vs. a number of satellite supplies or regulators sharing a common primary source (which might itself be a dc voltage derived from the ac mains). For most commercially-available modular converters, supply voltage used for operational amplifiers is usually adequately regulated to provide rated performance. Converter performance specifications as a function of dc variation of power supply voltage are almost always provided by the manufacturer.

DIGITAL LOGIC LEVELS

There are a variety of voltage levels and current-drive capacities corresponding to logic "0" and logic "1." This variety is a result of historical compromises between the need for speed, reliable differentiation between the logic states, circuit complexity, fanout capability, and the limitations of circuit/processing technology. They are described by such sets of initials as RTL, DTL, HTL, ECL, CMOS, IIL. Most of the modern modular conversion system products are designed to be compatible with TTL, which is the most widely-used logic system at present.

In TTL, as mentioned early in this chapter, a gate must respond to "0" if the input to it is 0.8 volts or less, and it must respond to "1" if the input is 2.0 volts or greater, up to the maximum and minimum voltage ratings. In order to provide a measure of immunity to noise, including dc voltage drops, occurring in transmission, gate outputs (within their current ratings), must furnish a minimum of 2.4 volts to signify "1" and a maximum of 0.4 volts to signify "0."

Within the TTL system, there are further classifications by fanout (the number of gates that can be driven) and speed. For convenience, input or output currents are normalized in terms of the standard *TTL load*, which is a positive current of 40μA for "1" and −1.6mA (sink current) for "0."

In addition to the *bipolar* transistor logic circuitry, *MOSFET* logic (e.g., CMOS) is often used. Because CMOS logic can operate at higher voltages, with greatly-increased noise immunity, and because of its low power consumption and accordingly higher circuit-packing density, it is especially useful in remote and portable system elements. The Analog Devices ADC1121 and DAC1122 are examples of converters specifically designed for low-power applications, using CMOS, but compatible with TTL. Examples of CMOS IC types include the AD7520 D/A converter and the AD7550 A/D converter.

Products designed for TTL logic can be used with other logic schemes by performing appropriate transformations (level-shifting, gain-or-attenuation, sign-inversion). D/A converters designed for TTL logic will inherently accept DTL inputs.

CONTROL LOGIC: THE *STATUS* OUTPUT

Most types of A/D Converters – except for those that perform continuous conversions – require a time interval, that may be either fixed or variable, to perform the conversion. During this time, the outputs may be changing and may bear no relationship to the final result. If a converter is interrogated during conversion, erroneous information will be transmitted. For this reason, the control output called *Status* (or "busy," "ready," etc.) changes state in response to the Convert Command to define the beginning of conversion and does not return to the original state until conversion is completed. It may be used to inhibit readout or to activate a *buffer* output register that holds the previous output word. It may also serve to prevent another conversion from beginning until the previous one is accomplished.

CONTROL LOGIC: THE *STROBE*

Most D/A converters – except for serial types, such as those that depend on charging of capacitors – have basic circuitry that responds immediately and continuously to whatever digital signals are applied. It is often desirable to isolate the basic circuitry from the source of digital information by a register, and update all bits simultaneously, upon command. The command input is called the *strobe* (or "clock" or "enable.")

For use with microprocessors, the data word is often divided into *bytes*, having a maximum of 8 bits (see Chapter I-4), which are *enabled* in sequence to transmit the information contained in the full word in *byte-serial* format. Input strobes to DAC's might be called *high-byte strobe* (more-significant bits), *low-byte strobe* (less-significant bits), and – to load the complete information into the DAC – *load-DAC strobe*. Conversely, the parallel outputs of ADC's are placed on the microprocessor data bus by *high-byte enable* and *low-byte enable* strobes. In order for the *status* output to be treated as information appearing on the data bus, a *status-enable* strobe would be used with microprocessor-compatible ADC's.

ANALOG SIGNALS

Inputs to A/D converters are usually in the form of voltage. Outputs from D/A converters are often in the form of voltage, at low impedance, from an operational amplifier (an example is AD564. However, many converters provide an output *current* instead of a voltage (for example, AD563). As will become clear in the sections that follow, the basic conversion process may inherently develop a current output that is quite fast, linear, and free from offset. A built-in operational amplifier may be used to convert that current to voltage, but at the present state-of-the-art, on-chip IC op amps having submicrosecond settling time to useful resolution are not available. As a result of the inevitable design tradeoffs, the amplifier will tend to limit converter performance, primarily by increasing settling time. If the current is made available directly, the speed of response is under the control of the user, through the choice of an appropriate external output amplifier. He can also choose the inverting or the noninverting mode. For example, the full-scale settling time of the current output from AD561 to 0.05% (½LSB of 10 bits) is 250 nanoseconds. The AD561, followed

by a general-purpose I.C. operational amplifier for voltage output, has settling time of 5μs to the same resolution; but with the high-speed AD509, it can be reduced to 600ns.

Converters that have current outputs or "soft" voltage outputs (directly from resistive ladders) may be considered as either voltage generators with series resistance or current generators with parallel resistance (Figure 11). They are used with operational amplifiers in either the inverting or the noninverting connection (Figure 12). Some types, such as the AD561 have one or more internal feedback resistors (for appropriate output voltage scaling) that track the ladder resistors, to minimize temperature variations of gain in inverting configurations. Also present may be a terminating resistor, to develop passively a noninverted output voltage, which may be amplified with a noninverting amplifier. The gain-determining feedback resistances (R_1, R_2) do not have to track the converter's internal resistors, only one another.

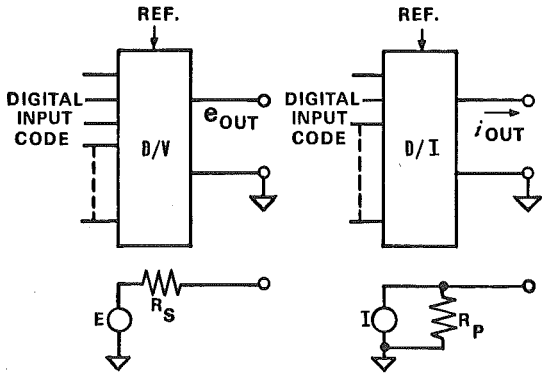


Figure 11. Digital-to-analog converters as voltage or current generators.

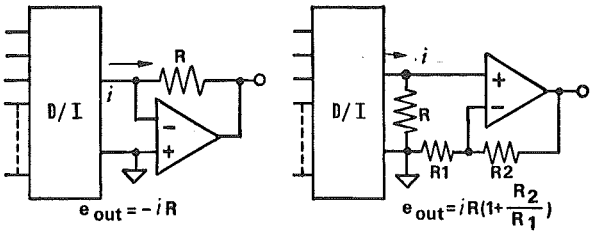


Figure 12. Current-to-voltage conversion using operational amplifier.

Using current-output converters, the inverting connection is the preferred connection, for a number of reasons: With current output, the internal impedance of the D/I converter is usually high. Thus, the loop gain will tend to remain near unity, essentially independently of the value of feedback resistance, minimizing amplifier-contributed errors, such as voltage drift. Furthermore, the output swing of the D/I converter (at the amplifier's negative input terminal) will be negligible, minimizing loading of the current output – and any associated problems, such as voltage-dependent nonlinearity and variation of internal impedance with temperature. Finally, common-mode rejection is not important, since there is no common-mode swing.

The conversion relationship of D/I converters is "positive reference" (Fig. 6) if the current flowing *out of the converter* increases as the value represented by the digital code increases, irrespective of the actual polarity of the converter's reference element. If a noninverting amplifier configuration is used, the output voltage will have the same normalized conversion relationship as the output current. If an inverting connection is used, the voltage will have a conversion relationship of opposite output polarity. Figure 13 illustrates this point, for both binary and complementary binary unipolar codes. On the other hand, if current flowing *towards the converter* increases as the value represented by the digital code increases, the relationship is "negative reference" for current, but "positive reference" for voltage in an inverting configuration.

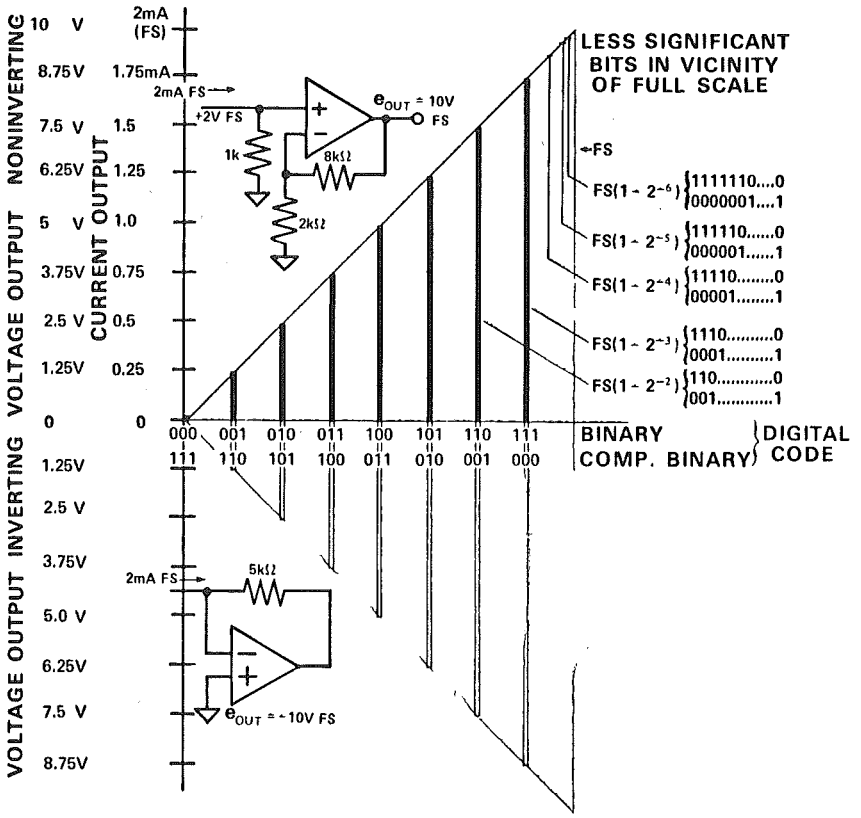


Figure 13. Ideal conversion relationships for 3 most-significant bits in positive reference unipolar D/I converter with noninverting and inverting amplifier connections, and binary vs. complementary binary codes.

D/A CONVERTER CIRCUITS

A basic D/A converter consists of a reference, a set of binary-weighted precision resistors, and a set of switches (Figure 14).

In this example, an operational amplifier holds one end of all the resistors at zero volts. The switches are operated by the digital logic, open for "0," closed for "1." Each switch that is closed adds a binary-weighted increment of current E_{REF}/R_j via the summing bus connected to the amplifier's negative input. The negative output voltage is proportional to the total current, and thus to the value of the binary number.

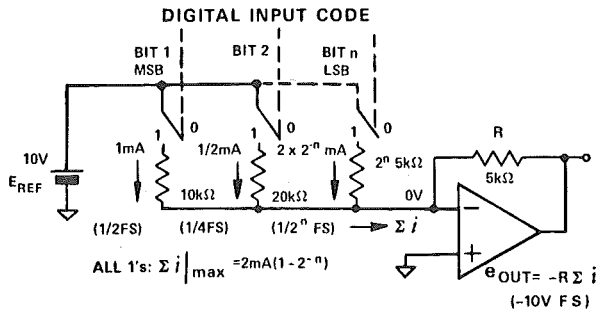


Figure 14. Simple D/A converter.

In practical applications, say for 12-bit D/A conversion, the range of resistance values would be 4,096:1, or 20MΩ for the LSB. If the resistors are to be manufactured in thin- or thick-film, or integrated-circuit form, such a range would be totally impractical. If discrete resistors are used, cost and size are increased, tracking advantages are lost, and inventory becomes a problem

Resistance Ladders

A way to reduce the resistance range is to use a limited number of repeated values, with suitable attenuation. One convenient approach, shown in Figure 15, is to use a binary resistance quad, consisting of the first four values (i.e. 2R, 4R, 8R, 16R) for each group of 4 bits, with attenuation of 16:1 for the second quad, 256:1 for the third quad, etc. As an additional benefit, the proper relative quad weighting for BCD conversion is achieved by changing the attenuation between quads to 10:1.

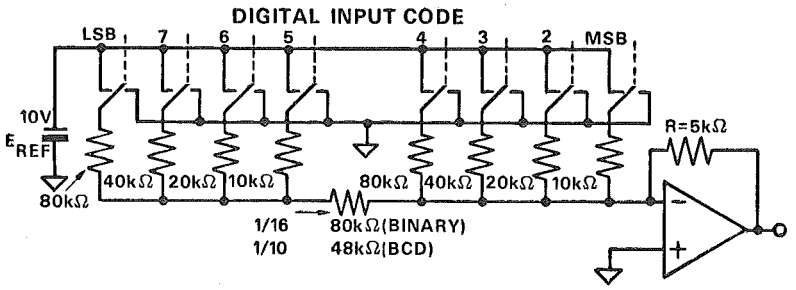
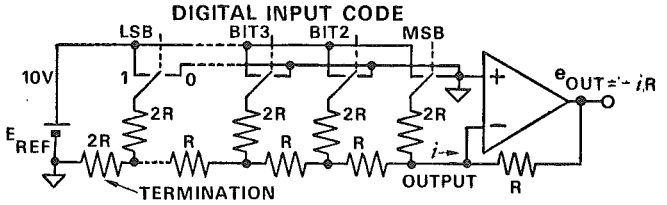
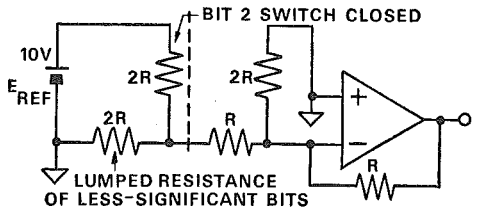


Figure 15. 8-Bit D/A converter using two equal-resistance quads with attenuation for the less-significant quad.

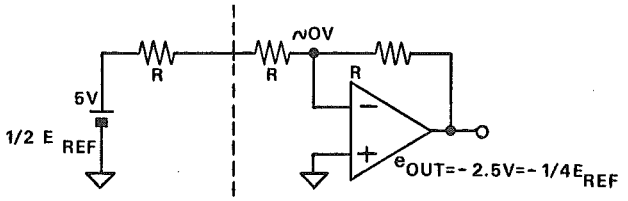
Carrying this reduction of resistance values all the way, one arrives at the R-2R ladder, another convenient – and very popular – form, depicted in Figure 16, which shows its use with an inverting operational amplifier. If all bits but the MSB are "off" (i.e., grounded), the output voltage is (-R/2R) E_{REF}. If all bits but bit 2 are off, it can be shown that the output voltage is $\frac{1}{2}(-R/2R)E_{REF} = \frac{1}{4}E_{REF}$: The lumped resistance of all the less-significant-bit circuitry (to the left of Bit 2) is 2R; the Thévenin equivalent looking back from the MSB towards Bit 2 is the generator, $E_{REF}/2$, and the series resistance $2R$; since the grounded MSB series resistance, $2R$, has virtually no influence – because the amplifier summing point is at virtual ground – the output voltage is therefore $-E_{REF}/4$. The same line of thinking can be employed to show that the nth bit produces an increment of output equal to $2^{-n}E_{REF}$.



a. Basic circuit.



b. Example: Contribution of bit 2; All other bits "0".



c. Simplified equivalent of circuit (b).

Figure 16. D/A converter using R-2R ladder network in current mode.

The R-2R network can be employed to give unattenuated noninverting output simply by connecting the output terminal to a high-impedance load, such as the input of a follower-connected operational amplifier (Figure 17). If the line of reasoning described in the preceding paragraph is followed, it can be seen that the MSB output is $\frac{1}{2}E_{REF}$ (2R-2R divider), Bit 2 is $\frac{1}{4}E_{REF}$ ($\frac{1}{2}E_{REF}$ equivalent generator and 2R-2R divider), etc. Since the entire network may be considered to be an equivalent generator having an output voltage NE_{REF} where N is the fractional digital number) and an internal resistance R, the output may be scaled down accurately by connecting precise resistance values to ground. Because of its symmetry and self-duality, the R-2R network may be used in other configurations. Figure 18 shows one example, in which the input and output leads, as depicted in Figure 16, are interchanged, for use in *current*-switching conversion.

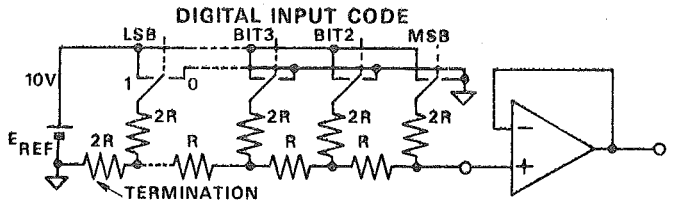


Figure 17. D/A converter using R-2R ladder network in voltage mode.

Switching

A thorough description of the variety of voltage and current switches actually used in converters would be beyond the scope of this chapter. However, the use of monolithic quad switches in converter design is discussed in Chapter II-3. Information on typical CMOS voltage switch quads will be found in the Analog Devices short-form *Product Guide.* Voltage switches are used in converters such as the AD7520 in the manner indicated in Figure 18, switching between the reference and ground, in order to maintain constant impedance. Since they accept reference voltage of either polarity, the AD7520's may be used as 4-quadrant multiplying DAC's.

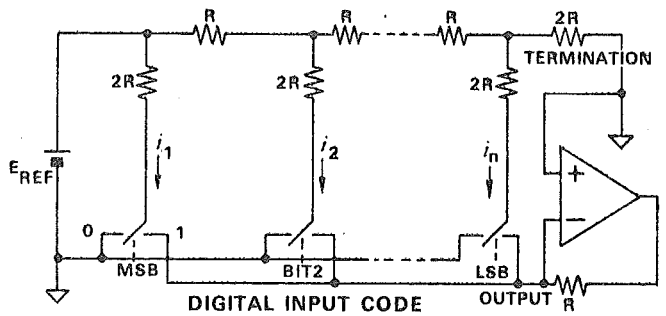


Figure 18. Inverted R-2R ladder in current-switching mode.

Current switches are used to steer current between an amplifier summing point and ground. They are capable of considerably higher speeds than voltage switches, because the reference current is not interrupted, and the only significant voltage changes appear at the output, (in response to code changes) but not across the switches. A simplified form of current switch, first employed in the '60's-vintage MDA-U Minidacs®, is shown in Figure 19. In this scheme, the switching transistors, in effect, isolate the weighting resistors from the output line (and its attenuators).

To understand the switching scheme of Figure 19, consider the transistor Q1. If its base-emitter voltage is equal to that of Q_R, the voltage across 2R will be equal to E_{REF}. The current through the resistor will therefore be $\frac{1}{2}E_{REF}/R$, and (assuming no current through the diode, and very high β) this same current will flow through to the collector circuit (i.e., the output). In order for this to happen, the cathode of the diode must be above the anode potential (i.e.,logic "1.")

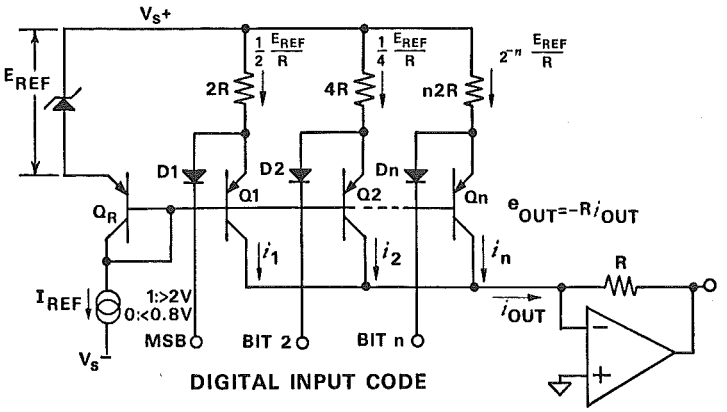


Figure 19. D/A converter using basic current switching technique.

If the base line is at 1.4V, and assuming 0.6V diode conduction voltage, an anode voltage of 2V (minimum for TTL logic "1") would be sufficient for this condition. If the anode of D1 is now switched to "0" (0.8V or less), Q1 will be cut off, because D1 will steal all its current, clamping the emitter of Q1 at or below the base potential, and eliminating Q1's contribution to the output. Since the current is not interrupted, and the voltage change is small, switching time is quite short, with settling to within 1 LSB typically of the order of 200 nanoseconds.

If Q1 and Q_R are matched for V_{BE} and have equal currents flowing through their emitter circuits, the voltage across 2R will track E_{REF} with temperature, making the MSB current essentially independent of temperature, except for β variations. The lesser-order bits operate in similar manner, except that the switching transistors are matched to the reference at the appropriate binary-weighted current levels. Tracking with temperature at these levels is adequate.

The switches and resistors are grouped in quads, with repeated 2R, 4R, 8R, 16R resistance values and 8:1 maximum range of bit currents. The less-significant bit currents are attenuated in the output line (Figure 20). An important benefit of the quad structure is that there are only four different current values, considerably reducing the dimensions of the current-matching problem, and maintaining adequate switching speed for the less-significant bits. Because of the attenuation, the tolerances on the resistor values and transistor tracking in the less-significant quads are greatly relaxed relative to the most significant quad. In monolithic quads, e.g. those in the AD562, the switching transistors have emitter areas bearing power-of-two relationships, so as to inherently maintain constant current density and hence equal (and tracking) V_{BE} and β. The reference transistor, on the same chip, is identical to one of the switching transistors.

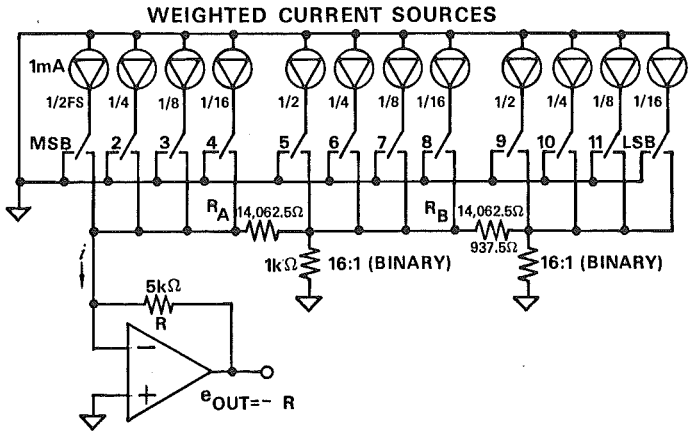


Figure 20. Block diagram of 12-bit current-switching D/A converter, showing weighted attenuation of quad output currents for binary coding. For BCD, R_A and R_B become 8,132.5Ω and 8,437.5Ω, and R becomes 4kΩ, and the interquad attenuations are 10:1.

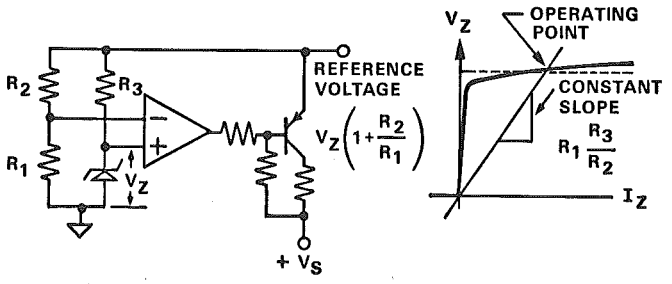
References

The most widely-used reference device is the temperature-compensated zener diode, often used with operational amplifiers for operating-point stabilization, unloading, or transducing to current (Figure 21). Some useful reference-circuit designs are discussed in chapter II-3. Complete references are also available in IC form and are either of the zener type (AD2700, 10V) or band-gap type (AD580, 2.5V).[6]

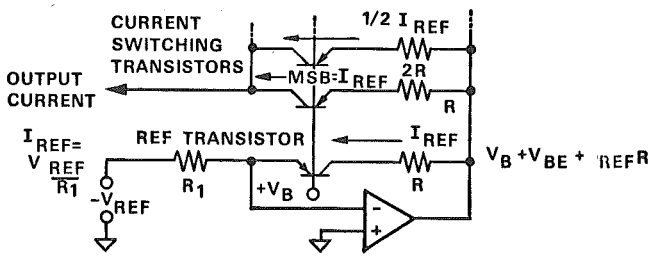
Bipolar Conversion

For bipolar current-switching D/A conversion, using offset binary or two's complement codes, an offset current equal and opposite to the MSB current is added to the converter output. This may be accomplished with a resistor and a separate offset reference, but more usually, it is derived from the converter's basic reference, in order to minimize drift of the output zero with temperature.

The gain of the output inverting amplifier must be doubled, in order to double the output range, e.g., from 0 - 10V to ±10V. As indicated earlier, (Figure 6) zero output corresponds to offset binary 1 0 0 . . . 0 0, or two's complement 0 0 0 . . . 0 0.



a. Amplifier adjusts feedback current to stabilize zener operating point independently of V_S or load variations.



b. Amplifier converts reference voltage to reference current in current-switching converter.

Figure 21. Examples of use of operational amplifiers in generating reference voltage and current.

Figure 22 shows an example of a current-switching converter connected for bipolar output. Note that because the amplifier is connected for sign inversion, the overall conversion relationship is "negative reference," i.e., +F.S. for all 0's (offset binary), -F.S. (1 - LSB) for all 1's.

[6] *Analog Dialogue* 9-1 (theory) and 9-2 (applications), 1975.

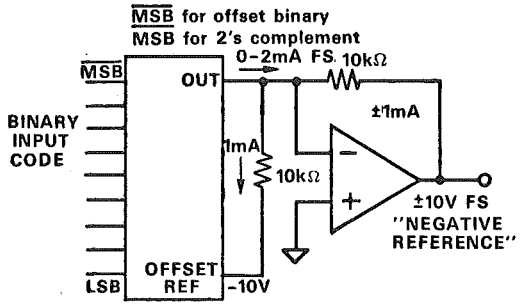


Figure 22. Bipolar connection of current-swtiching D/A converter for offset binary or 2's complement codes.

For non-inverting applications, the same values of offset voltage and resistance are used, but the proper value of output voltage scale factor depends on the load presented by the parallel combination of the internal resistance, the offset resistance, and the external load. (Figure 23)

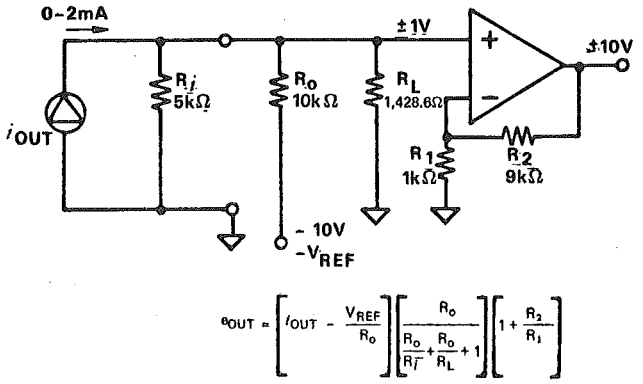


Figure 23. Non-inverting output from current-switching D/A converter.

For bipolar D/A conversion using the voltage switches and R-2R ladder network of Figures 16 and 17, and offset binary or 2's complement coding, one approach is to drive those network terminals that are normally grounded for unipolar operation (one side of the switches and the LSB termination) with the reference signal in the opposite polarity. If the LSB termination is allowed to remain grounded, the output will be symmetrical, resembling the conversion relationship of Figure 8, with no code for analog zero but with normalized gain reduced to $1 - 2^{-(n+1)}$

For sign-magnitude conversion, the converter's current output may be inverted, using a current inverter. Switch circuitry, operated by the MSB, determines whether the output amplifier's input is direct or via the current inverter. One way of accomplishing this is shown in Figure 24.

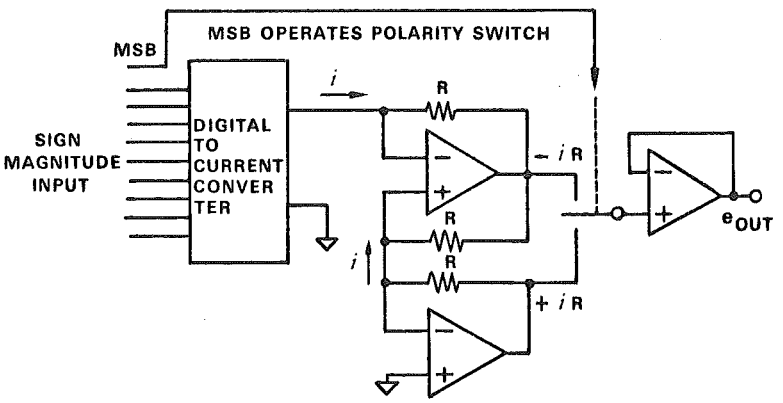


Figure 24. Sign-magnitude bipolar D/A conversion using unipolar DAC and switched outputs of operational amplifiers.

Registers on DAC's (Figure 25)

The basic parallel-input D/A converter circuits considered so far have the common property that the analog output continually reflects the state of the logic inputs. If the basic conversion circuitry is preceded by a register, the device will respond only to the inputs gated into it. This property is especially useful in data distribution systems, in which data is continually appearing, but it is desired that a DAC respond only at certain times, then hold the analog output constant until the next update. In this sense, a DAC with buffer storage may be viewed as a sample-hold with digital input, analog output, and (conceivably) infinite "Hold" time.

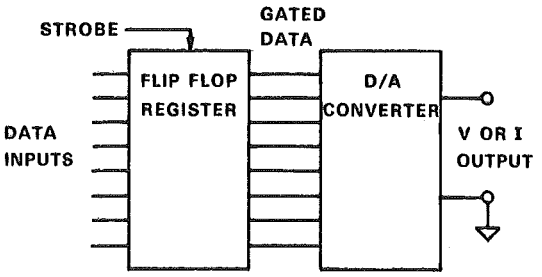


Figure 25. D/A converter with buffer register.

The register is controlled by a *strobe* signal, which causes it to update. The limiting rate at which the strobe may update is determined by two factors: the settling time of the DAC, and the response time of the logic. In general, settling time of the analog portion of the D/A converter is at least an order of magnitude slower than the response time of modern high-speed TTL logic circuits and is thus the limiting factor on update rate.

The only time when the speed of the digital portion of a D/A converter is of importance is when the "glitch" (See II-5) caused by unequal turn-on and turn-off times is an important factor in the application. The digital inputs to a DAC come from digital logic circuits, which exhibit *skew*, or unequal turn-on and turn-off times. The switches used in DAC's also exhibit skew; however, even if the switch circuitry is specifically designed to minimize skew, the additional skew of the digital logic will constitute an irreducible minimum. In such circuit applications, glitch energy can be minimized by using high-speed logic.

A/D CONVERTER CIRCUITS

There are a vast number of conceivable circuit designs for A/D converters.[7] There are a much more limited number of designs available on the market in small, modular form at low cost, specifically designed for incorporation as components of equipment. The most popular of these are :

Successive-approximation types
Integration (single-, dual-, and quad-slope) and V-to-f types
Counter and "servo" types
Parallel and modified-parallel types

Each approach has characteristics that make it most useful for a specific class of applications, based on speed, accuracy, cost, size, versatility.

Successive-Approximation

Successive-approximation A/D converters are quite widely used, especially for interfacing with computers, because they are capable of both high resolution (to 16 bits: ADC-16Q), and high speed (to 1 MHz throughput rates: ADC-1103). Conversion time is fixed and independent of the magnitude of the input voltage. Each conversion is unique and independent of the results of previous conversion, because the internal logic is cleared at the start of a conversion.

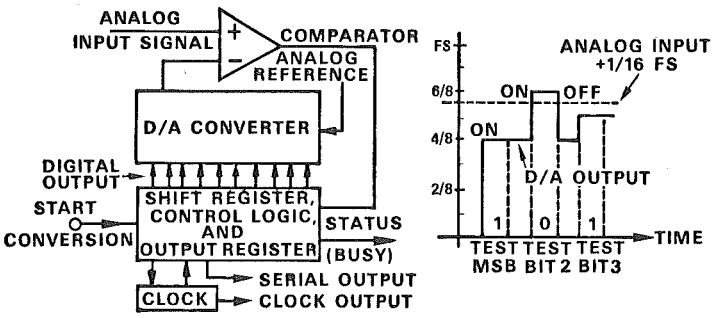
[7] See *Electronic Analog/Digital Conversions*, by H. Schmid (Van Nostand Reinhold, 1970) for an encyclopedic panoply of A/D (and D/A) converter circuit designs.

Modern IC converters, such as the monolithic AD7570 10-bit ADC, include 3-state data outputs and byte controls to facilitate interfacing with microprocessors. A "three-state" output has, in addition to the normal "1" and "0" states, when enabled, a not-enabled condition, in which the output is simply disconnected via an open voltage switch. This permits many device outputs to be connected to the same bus – only the device that is enabled (one at a time) can drive the bus. Since typical processor data buses are only 8 bits wide, 10- or 12-bit data must be communicated in two steps, one "byte" at a time.

The conversion technique consists of comparing the unknown input against a precisely-generated internal voltage at the output of a D/A converter. The input of the D/A converter is the digital number at the A/D converter's output. The conversion process is strikingly similar to a weighing process using a chemist's balance, with a set of *n* binary weights (e.g., 1/2lb, 1/4lb, 1/8lb, 1/16lb (= 1oz), 1/2oz, 1/4oz, etc., for unknowns up to 1 lb.)

After the conversion command is applied, and the converter has been cleared, the D/A converter's MSB output (1/2 full scale) is compared with the input. If the input is greater than the MSB, it remains ON (i.e., "1" in the output register), and the next bit (1/4FS) is tried. If the input is less than the MSB, it is turned OFF (i.e., "0" in the output register), and the next bit is tried. If the second bit doesn't add enough weight to exceed the input, it is left ON ("1"), and the third bit is tried. If the second bit tips the scales too far, it is turned OFF ("0"), and the third bit is tried. The process continues in order of descending bit weight until the last bit has been tried. The process completed, the *status* line changes state to indicate that the contents of the output register now constitute a valid conversion. The contents of the output register form a binary digital code corresponding to the input signal.

Figure 26 is a block diagram of a successive-approximations A/D converter, accompanied by a time history of a simple 3-bit conversion, in terms of the D/A converter output (the weight added to the balance pan). Note that, to place the D/A converter output in the center of each ideal output quantum, a 1/2-LSB "thumb" is placed on the scale (see Figure 2, this chapter), in order to locate the transitions precisely at the 1/2LSB points.



a. Block Diagram *b. 3-Bit Weighing*

Figure 26. Successive-approximation A/D converter.

Note that the input does not change during conversion in the example of Figure 26b. If the input were to change during conversion, the output number could no longer accurately represent the analog input unless the new value were larger than the sum of the weights already present by an amount less than the sum of the untried weights. Since this is not-often-fulfilled requirement, it is usual to employ a sample-hold device ahead of the converter to retain the input value that was present at a given time before the conversion starts, and maintain it constant throughout the conversion. The *status* output of the converter could be used to release the sample-hold from its *hold* mode at the end of conversion. A sample-hold may not be needed if the signal (by itself, or with filtering) varies slowly enough and is sufficiently noise-free that significant changes will not be expected to occur during the conversion interval.

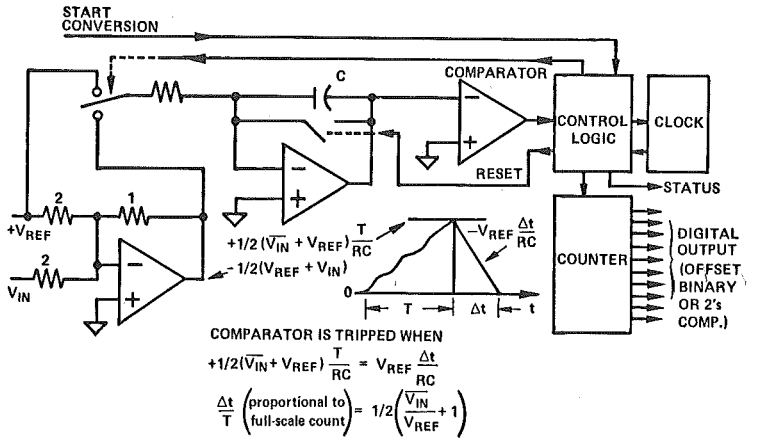
Accuracy, linearity, and speed are primarily affected by the properties of the D/A converter (and its reference), and the comparator. In general, the settling time of the D/A converter and the response time of the comparator are considerably slower than the switching time

of the digital elements. The differential nonlinearity of the D/A converter will be reflected in the differential nonlinearity of the resulting A/D converter. If the D/A converter is non-monotonic, one or more codes may be missing from the A/D converter's output range. Bipolar inputs are dealt with by using a D/A converter with bipolar output and offset binary coding, and appropriate input scaling.

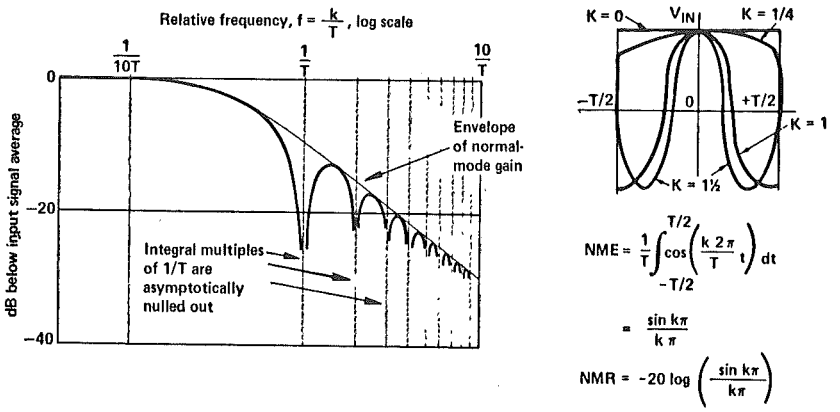
Integration (Ramp and V-to-f Types)

This family of converters is also quite popular. Its members perform an *indirect* conversion, by first converting to a function of time, then converting from the time function to a digital number using a counter. The *dual-ramp type* is especially suitable for use in digital voltmeters and those applications in which a relatively-lengthy time may be taken for conversion to obtain the benefits of noise reduction through signal averaging.

Here's how the dual-ramp type works: The input signal is applied to an integrator; at the same time a counter is started, counting clock pulses. After a predetermined number of counts (a fixed interval of time,T), a reference voltage having opposite polarity is applied to the integrator. At that instant, the accumulated charge on the integrating capacitor is proportional to the average value of the input over the interval T. Th ntegral of the reference is an opposite-going ramp having a slope V_{REF}/RC. At the same time, the counter is again counting from zero. When the integrator output reaches zero, the count is stopped, and the analog circuitry is reset. Since the charge gained is proportional to $\overline{V_{IN}}$ T, and the equal amount of charge lost is proportional to $V_{REF}\Delta t$, then the number of counts relative to the full count is proportional to $\Delta t/T$, or $\overline{V_{IN}}/V_{REF}$. If the output of the counter is a binary number, it will therefore be a binary representation of the input voltage. If the input is attenuated and offset by half the reference voltage, the output will be an offset binary representation of a bipolar input, suitable as an input for computer systems. Figure 27a shows a dual-ramp A/D converter for bipolar signals with offset-binary output.



a. Dual-ramp A/D converter for bipolar input.



b. Worst-cast normal-mode response of dual-slope A/D converter.

Figure 27. Voltage-to-time-to-digital converters.

Dual-slope integration has many advantages. Conversion accuracy is independent of both the capacitor value and the clock frequency, because they affect both the up-slope and the down-ramp in the same ratio. Differential linearity is excellent, because the analog function is free from discontinuities, the codes are generated by a clock and counter, and all codes can inherently exist. Resolution is limited only by analog resolution, rather than by differential nonlinearity; hence, the excellent fine structure may be represented by more bits than would be needed to maintain a given level of scale-factor accuracy. The integration provides rejection of high-frequency noise and averaging of changes that occur during the sampling period. The fixed averaging period also makes it possible to obtain "infinite" normal-mode rejection[8] at frequencies that are integral multiples of $1/T$ (see Figure 27b).

Throughput rate of dual-slope converters is limited to somewhat less than $1/2T$ conversions per second. The sample time, T, is determined by the fundamental frequency to be rejected. For example, if one wishes to reject 60Hz and its harmonics, the minimum integrating time is 16-2/3 ms, and the maximum number of conversions is somewhat less than 30/s. Though too slow for fast data acquisition, dual-slope converters are quite adequate for such transducers as thermocouples and gas chromatographs; and they are the predominant circuit used in constructing digital voltmeters. Since DVM's use sign-magnitude BCD coding, bipolar operation requires polarity sensing and reference-polarity switching, rather than simple offsetting.

A shortcoming of conventional dual-slope converters is that errors at the input of the integrating amplifier or the comparator show up as errors in the digital word. Such errors are usually reduced by the introduction of a third portion of the cycle, during which a capacitor is charged with zero-drift errors, which are then introduced in the opposite sense during the integration to (it is hoped) nullify them. An interesting scheme (applied to the 13-bit single-chip AD7550) discussed in Chapter II-2, for nullifying all such input errors, is the patented *quad-slope* principle;* it stores the errors in the form of a digital count during a calibration cycle and subtracts them from the final count during the conversion cycle.

Other conversion approaches in this class include the single-ramp type and V/f converters. In the single-ramp converter, a reference voltage, of opposite polarity to the signal, is integrated (while a counter counts clock pulses) until the output of the integrator is equal to the signal input. At that time (Δt) the output of the integrator is $E_{REF}\Delta t/RC$. Therefore, Δt – hence, the number counts and the corresponding digital number – is proportional to the ratio of the input to the reference. This process has the weakness that its accuracy depends on both the capacitor (extremely-accurate and stable resistors are relatively easy to come by) and the clock frequency. In the V/f converter, a frequency is generated in proportion to the input signal; a counter measures the frequency and provides a digital output code, the value of which is proportional to the input signal. In both of the above schemes, offsetting may be used to obtain offset binary representation of bipolar analog inputs.

Counter and "Servo" Types

Figure 28 is a block diagram of a counter-comparator A/D converter, which is analogous to the single-ramp type, but it is indpendent of the time scale. The analog input is compared with the output of a D/A converter, the digital input of which is driven by a counter. At the start of the conversion, the counter starts its count, which continues until the D/A output crosses the input value. At that point, conversion ceases, and the converter is ready to perform the next conversion after the counter has been cleared and its output dumped into a storage register. The number of counts appears in the output register. For bipolar inputs, a bipolar D/A converter is used, and the count is an offset binary representation of the input, starting from negative full-scale.

[8] *Normal-mode* noise consists of unwanted signals that appear on the input line, even if common-mode error is nil. If a low-frequency or dc quantity is to be converted in the presence of a high-frequency ripple, a successive-approximations A/D converter, even if preceded by a sample-hold, will convert the instantaneous values of signal-plus-noise, producing a noisy digital signal. On the other hand, an integrator will inherently attenuate high frequencies, producing smoothing, and, if combined with a fixed averaging period, will null out those frequencies that have whole numbers of cycles during the averaging period.

*U.S. Patent 3,872,466

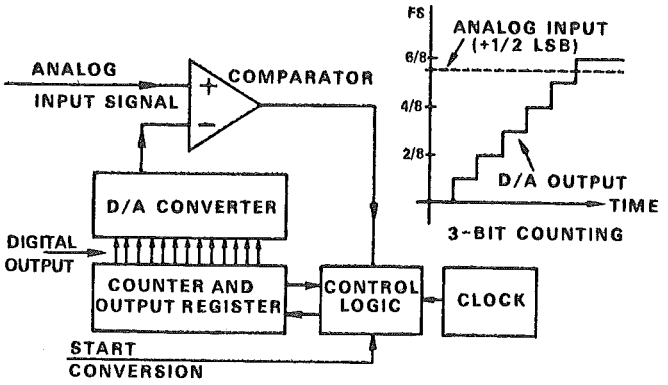


Figure 28. Counter-comparator A/D converter.

Though quite simple in concept, this converter has the disadvantage of limited speed for a given resolution, since the conversion time for a full-scale change is equal to the clock frequency divided into the maximum number of counts. For example, if the clock frequency is 10MHz, the maximum throughput rate for 10-bit resolution (1024 counts) is something less than 10kHz (100μs per conversion). A variation of this converter is the "servo" type, in which an "up-down" counter is used.

If the output of the D/A converter is less than the analog input, the counter counts up. If the D/A output is greater than the analog input, the counter counts down. If the analog input is constant, the counter output "hunts" back and forth between the two adjacent bit values. This converter can follow small changes quite rapidly (it will follow 1 LSB changes at the clock rate), but it will require the full count to acquire full-scale step changes. Since it seeks to "home in" on the analog value, the analogy to a servomechanism is quite evident. It seeks to convert continuously, which may be a disadvantage in tying it in with a fast data-acquisition system, since it can give a valid "conversion complete" report only during the clock period immediately following a change in state of the comparator (which in general occurs at irregular intervals). A buffer storage register may be used to store the previous count, while the counter is seeking the next value. By stopping the count (following a completed conversion) at an externally-determined instant, the servo-type converter may be used as a sample-hold with arbitrarily-long *hold* time (with no droop). If the "up" or the "down" count is disabled, the converter will act as a valley follower or a peak follower, counting in the appropriate direction only when the analog input exceeds the previous extreme value. Both the analog and the digital stored values are available.

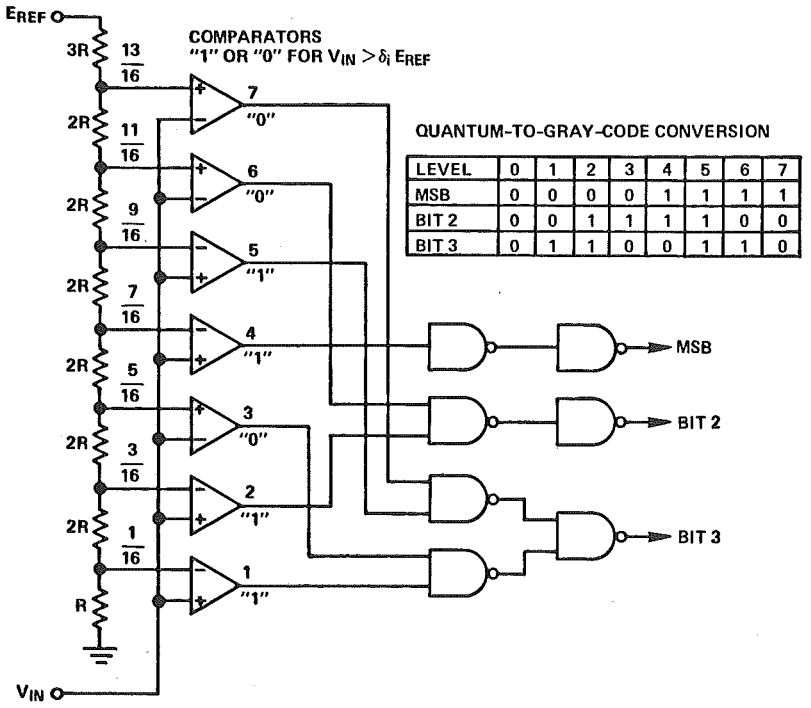
Parallel Types

Figure 29 shows a parallel 3-bit converter with Gray code output. It has $2^n - 1$ comparators, biased 1 LSB apart, starting with +1/2 LSB. For 0 input, all comparators are off. As the input increases, it causes an increasing number of comparators to switch state. The outputs of the comparators are applied to the gates, which provide a set of outputs that fulfill the appropriate conditions for Gray-code output. (Natural binary could be implemented in the same way, using an appropriate table).

The evident advantage of this approach is that conversion occurs in parallel, with speed limited only by the switching time of the comparators and gates. As the input changes, the output code changes. Thus, this is the fastest approach to conversion.

Unfortunately, the number of elements increases geometrically with resolution. As linear and digital integrated-circuit elements of increasing complexity become available, increased levels of resolution will tend to approach the threshold of practicality. But high resolution and the fastest speeds at low cost are still some time away.

By combining parallel conversion for small numbers of bits with iteration (to wit, successive approximations taking several bits at a time), it is possible to strike a compromise that gives better resolution than a parallel approach, with less complexity, and improved speed over the successive-approximations approach.



LEVEL	0	1	2	3	4	5	6	7
MSB	0	0	0	0	1	1	1	1
BIT 2	0	0	1	1	1	1	0	0
BIT 3	0	1	1	0	0	1	1	0

Figure 29. Parallel 3-bit A/D converter with Gray code output.

A NOTE ON SHARED LOGIC

In this book, we are concerned with the embodiment of the conversion function by means of IC's or other modules, in essentially complete form, with *completely defined specifications.* We should acknowledge, however, that in the consideration of the tradeoffs between hardware and software, a software-oriented designer will be tempted to consider hardware savings inherent in using the control-logic capability of a microprocessor, along with the precision analog function (the reference and the comparator – and the integrator or the DAC), to perform single or multiplexed conversions, employing the techniques mentioned here, but without using a piece of hardware identifiable as an "a/d converter" *per se.*

A decision to do this is in some respects equivalent to a decision to *design* a converter (analogous to the classical "make-or-buy" decision). While there are applications for which the approach is eminently fruitful (e.g., dedicated instruments, to be manufactured in large quantity), the usual tradeoffs should be considered, lest the fascination of expending design- and manufacturing-effort and software development in areas peripheral to one's primary mission, lead one down the "primrose path" of wasted resources.

CONCLUSION

In this chapter, we have attempted to provide the fundamentals for a basic understanding of converters. In the chapters that follow, we will discuss further some of the considerations faced by the converter designer, provide an understanding of and a guide to specification of converters, and explore the elements of successful system design using converters.