$2^4 = 16$. A four-bit group is called a *nibble*. Because hex requires 16 digits, the letters “A” through “F” are borrowed for use as hex digits beyond 9. The 16 hex digits are defined in Table 1.7.

TABLE 1.7 Hexadecimal Digits

Decimal value	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex digit	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Binary nibble	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

The preceding example, $191_{10} = 10111111_2$, can be converted to hex easily by grouping the eight bits into two nibbles and representing each nibble with a single hex digit:

$$1011_2 = (8 + 2 + 1)_{10} = 11_{10} = B_{16}$$

$$1111_2 = (8 + 4 + 2 + 1)_{10} = 15_{10} = F_{16}$$

Therefore, $191_{10} = 10111111_2 = BF_{16}$. There are two common prefixes, 0x and \$, and a common suffix, h, that indicate hex numbers. These styles are used as follows: BF_{16} = 0xBF = \$BF = BFh. All three are used by engineers, because they are more convenient than appending a subscript “16” to each number in a document or computer program. Choosing one style over another is a matter of preference.

Whether a number is written using binary or hex notation, it remains a string of bits, each of which is 1 or 0. Binary numbering allows arbitrary data processing algorithms to be reduced to Boolean equations and implemented with logic gates. Consider the equality comparison of two four-bit numbers, M and N.

“If M = N, then the equality test is true.”

Implementing this function in gates first requires a means of representing the individual bits that compose M and N. When a group of bits are used to represent a common entity, the bits are numbered in ascending or descending order with zero usually being the smallest index. The bit that represents 2^0 is termed the *least-significant bit*, or LSB, and the bit that represents the highest power of two in the group is called the *most-significant bit*, or MSB. A four-bit quantity would have the MSB represent 2^3. M and N can be ordered such that the MSB is bit number 3, and the LSB is bit number 0. Collectively, M and N may be represented as M[3:0] and N[3:0] to denote that each contains four bits with indices from 0 to 3. This presentation style allows any arbitrary bit of M or N to be uniquely identified with its index.

Turning back to the equality test, one could derive the Boolean equation using a variety of techniques. Equality testing is straightforward, because M and N are equal only if each digit in M matches its corresponding bit position in N. Looking back to Table 1.3, it can be seen that the XNOR gate implements a single-bit equality check. Each pair of bits, one from M and one from N, can be passed through an XNOR gate, and then the four individual equality tests can be combined with an AND gate to determine overall equality,

$$Y = \overline{M[3] \oplus N[3]} \& \overline{M[2] \oplus N[2]} \& \overline{M[1] \oplus N[1]} \& \overline{M[0] \oplus N[0]}$$

The four-bit equality test can be drawn schematically as shown in Fig. 1.8.

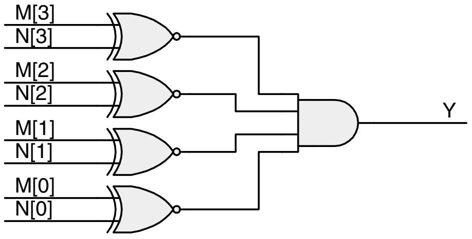

FIGURE 1.8 Four-bit equality logic.

Logic to compare one number against a constant is simpler than comparing two numbers, because the number of inputs to the Boolean equation is cut in half. If, for example, one wanted to compare M[3:0] to a constant 1001_2 (9_{10}), the logic would reduce to just a four-input AND gate with two inverted inputs:

$$y = M[3]\&\overline{M[2]}\&\overline{M[1]}\&M[0]$$

When working with computers and other digital systems, numbers are almost always written in hex notation simply because it is far easier to work with fewer digits. In a 32-bit computer, a value can be written as either 8 hex digits or 32 bits. The computer's logic always operates on raw binary quantities, but people generally find it easier to work in hex. An interesting historical note is that hex was not always the common method of choice for representing bits. In the early days of computing, through the 1960s and 1970s, *octal* (base-8) was used predominantly. Instead of a single hex digit representing four bits, a single octal digit represents three bits, because $2^3 = 8$. In octal, $191_{10} = 277_8$. Whereas bytes are the *lingua franca* of modern computing, groups of two or three octal digits were common in earlier times.

Because of the inherent binary nature of digital systems, quantities are most often expressed in orders of magnitude that are tied to binary rather than decimal numbering. For example, a "round number" of bytes would be 1,024 (2^{10}) rather than 1000 (10^3). Succinct terminology in reference to quantities of data is enabled by a set of standard prefixes used to denote order of magnitude. Furthermore, there is a convention of using a capital B to represent a quantity of bytes and using a lowercase b to represent a quantity of bits. Commonly observed prefixes used to quantify sets of data are listed in Table 1.8. Many memory chips and communications interfaces are expressed in units of bits. One must be careful not to misunderstand a specification. If you need to store 32 MB of data, be sure to use a 256 Mb memory chip rather than a 32 Mb device!

TABLE 1.8 Common Binary Magnitude Prefixes

Prefix	Definition	Order of Magnitude	Abbreviation	Usage
Kilo	$(1{,}024)^1 = 1{,}024$	2^{10}	k	kB
Mega	$(1{,}024)^2 = 1{,}048{,}576$	2^{20}	M	MB
Giga	$(1{,}024)^3 = 1{,}073{,}741{,}824$	2^{30}	G	GB
Tera	$(1{,}024)^4 = 1{,}099{,}511{,}627{,}776$	2^{40}	T	TB
Peta	$(1{,}024)^5 = 1{,}125{,}899{,}906{,}842{,}624$	2^{50}	P	PB
Exa	$(1{,}024)^6 = 1{,}152{,}921{,}504{,}606{,}846{,}976$	2^{60}	E	EB