**SIGNED BINARY NUMBERS**

Positive integers (including zero) can be represented as unsigned numbers. However, to represent negative integers, we need a notation for negative values. In ordinary arithmetic, a negative number is indicated by a minus sign and a positive number by a plus sign. Because of hardware limitations, computers must represent everything with binary digits. It is customary to represent the sign with a bit placed in the leftmost position of the number. The convention is to make the sign bit 0 for positive and 1 for negative.

It is important to realize that both signed and unsigned binary numbers consist of a string of bits when represented in a computer. The user determines whether the number is signed or unsigned. If the binary number is signed, then the leftmost bit represents the sign and the rest of the bits represent the number. If the binary number is assumed to be unsigned, then the leftmost bit is the most significant bit of the number. For example, the string of bits 01001 can be considered as 9 (unsigned binary) or as +9 (signed binary) because the leftmost bit is 0. The string of bits 11001 represents the binary equivalent of 25 when considered as an unsigned number and the binary equivalent of -9 when considered as a signed number. This is because the 1 that is in the leftmost position designates a negative and the other four bits represent binary 9. Usually, there is no confusion in interpreting the bits if the type of representation for the number is known in advance. representation of negative numbers. The same procedures can be applied to the signed‐1’s‐complement system by including the end‐around carry as is done with unsigned numbers. The representation of the signed numbers in the last example is referred to as the *signed‐magnitude* convention. In this notation, the number consists of a magnitude and a symbol ( + or - ) or a bit (0 or 1) indicating the sign. This is the representation of signed

numbers used in ordinary arithmetic. When arithmetic operations are implemented in a computer, it is more convenient to use a different system, referred to as the *signedcomplement* system, for representing negative numbers. In this system, a negative number is indicated by its complement. Whereas the signed‐magnitude system negates a number by changing its sign, the signed‐complement system negates a number by taking its complement. Since positive numbers always start with 0 (plus) in the leftmost position, the complement will always start with a 1, indicating a negative number. The signed‐complement system can use either the 1’s or the 2’s complement, but the 2’s complement is the most common.

As an example, consider the number 9, represented in binary with eight bits. +9 is

represented with a sign bit of 0 in the leftmost position, followed by the binary equivalent of 9, which gives 00001001. Note that all eight bits must have a value; therefore, 0’s are inserted following the sign bit up to the first 1. Although there is only one way to represent +9, there are three different ways to represent -9 with eight bits:

signed‐magnitude representation: 10001001

signed‐1’s‐complement representation: 11110110

signed‐2’s‐complement representation: 11110111

In signed‐magnitude, -9 is obtained from +9 by changing only the sign bit in the leftmost position from 0 to 1. In signed‐1’s-complement, -9 is obtained by complementing all the bits of +9, including the sign bit. The signed‐2’s‐complement representation of -9 is obtained by taking the 2’s complement of the positive number, including the sign bit.

Table 1.3 lists all possible four‐bit signed binary numbers in the three representations.

The equivalent decimal number is also shown for reference. Note that the positive numbers in all three representations are identical and have 0 in the leftmost position. The signed‐2’s‐complement system has only one representation for 0, which is always positive.

The other two systems have either a positive 0 or a negative 0, something not

encountered in ordinary arithmetic. Note that all negative numbers have a 1 in the leftmost bit position; that is the way we distinguish them from the positive numbers.

With four bits, we can represent 16 binary numbers. In the signed‐magnitude and the 1’s‐complement representations, there are eight positive numbers and eight negative numbers, including two zeros. In the 2’s‐complement representation, there are eight positive numbers, including one zero, and eight negative numbers.

The signed‐magnitude system is used in ordinary arithmetic, but is awkward when

employed in computer arithmetic because of the separate handling of the sign and the magnitude. Therefore, the signed‐complement system is normally used. The 1’s complement imposes some difficulties and is seldom used for arithmetic operations. It is useful as a logical operation, since the change of 1 to 0 or 0 to 1 is equivalent to a logical complement operation, as will be shown in the next chapter. The discussion of signed binary arithmetic that follows deals exclusively with the signed‐2’s‐complement representation of negative numbers. The same procedures can be applied to the signed‐1’s‐complement system by including the end‐around carry as is done with unsigned numbers. 