

Table 2-5
Low-resolution graphics colors

Nibble value		
Dec	Hex	Color
0	\$00	Black
1	\$01	Magenta
2	\$02	Dark blue
3	\$03	Purple
4	\$04	Dark green
5	\$05	Gray
6	\$06	Medium blue
7	\$07	Light blue
8	\$08	Brown
9	\$09	Orange
10	\$0A	Gray 2
11	\$0B	Pink
12	\$0C	Light green
13	\$0D	Yellow
14	\$0E	Aquamarine
15	\$0F	White

Note: Colors may vary, depending upon the controls on the monitor or TV set.

Half a byte—four bits, or one nibble—is assigned to each graphics block. Each nibble can have a value from 0 to 15, and this value determines which one of 16 colors appears on the screen. The colors and their corresponding nibble values are shown in Table 2-5. In each byte, the low-order nibble sets the color for the top block of the pair, and the high-order nibble sets the color for the bottom block. Thus, a byte containing the hexadecimal value \$D8 produces a brown block atop a yellow block on the screen.

As explained later in the section “Video Display Pages,” the text display and the low-resolution graphics display use the same area in memory. Most programs that generate text and graphics clear this part of memory when they change display modes, but it is possible to store data as text and display it as graphics, or vice-versa. All you have to do is change the mode switch, described later in this chapter in the section “Display Mode Switching,” without changing the display data. This usually produces meaningless jumbles on the display, but some programs have used this technique to good advantage for producing complex low-resolution graphics displays quickly.

High-resolution graphics

In the high-resolution graphics mode, the Apple IIe displays an array of colored dots in 192 rows and 280 columns. The colors available are black, white, purple, green, orange, and blue, although the colors of the individual dots are limited, as described later in this section. Adjacent dots of the same color merge to form a larger colored area.

Data for the high-resolution graphics displays are stored in either of two 8192-byte areas in memory. These areas are called *high-resolution Page 1* and *Page 2*; think of them as buffers where you can put data to be displayed. Normally, your programs will use the features of some high-level language to draw graphics dots, lines, and shapes to display; this section describes the way the resulting graphics data are stored in the Apple IIe’s memory.

The Apple IIe high-resolution graphics display is bit-mapped: each dot on the screen corresponds to a bit in the Apple IIe’s memory. The seven low-order bits of each display byte control a row of seven adjacent dots on the screen, and forty adjacent bytes in memory control a row of 280 (7 times 40) dots. The least significant bit of each byte is displayed as the leftmost dot in a row of seven, followed by the second-least significant bit, and so on, as shown in Figure 2-5. The eighth bit (the most significant) of each byte is not displayed; it selects one of two color sets, as described later.

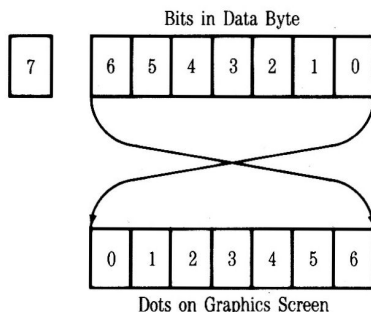


Figure 2-5
High-resolution display bits