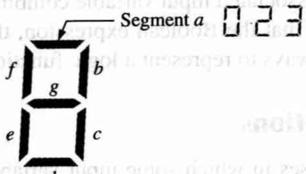
FIGURE 4–39

7-segment display.

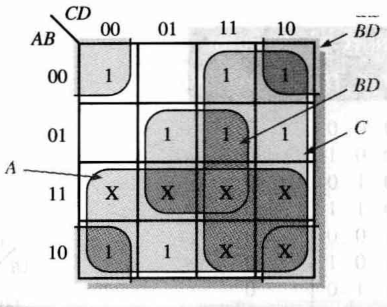


*Solution* The expression for segment *a* is

$$a = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,\overline{B}C\overline{D} + \overline{A}\,\overline{B}CD + \overline{A}B\overline{C}D + \overline{A}BC\overline{D} + \overline{A}BCD + A\overline{B}\,\overline{C}\,\overline{D} + A\overline{B}\,\overline{C}D$$

Each term in the expression represents one of the digits in which segment *a* is used. The Karnaugh map minimization is shown in Figure 4–40. X's (don't cares) are entered for those states that do not occur in the BCD code.

FIGURE 4–40



From the Karnaugh map, the minimized expression for segment *a* is

$$a = A + C + BD + \overline{B}\,\overline{D}$$

*Related Problem* Draw the logic diagram for the segment-*a* logic.

**SECTION 4–9 CHECKUP**

1. Lay out Karnaugh maps for three and four variables.
2. Group the 1s and write the simplified SOP expression for the Karnaugh map in Figure 4–27.
3. Write the original standard SOP expressions for each of the Karnaugh maps in Figure 4–34.

## 4–10 FIVE-VARIABLE KARNAUGH MAPS

Boolean functions with five variables can be simplified using a 32-cell Karnaugh map. Actually, two 4-variable maps (16 cells each) are used to construct a 5-variable map. You already know the cell adjacencies within each of the 4-variable maps and how to form groups of cells containing 1s to simplify an SOP expression. All you need to learn for five variables is the cell adjacencies between the two 4-variable maps and how to group those adjacent 1s.

After completing this section, you should be able to

- Determine cell adjacencies in a 5-variable map
- Form maximum cell groupings in a 5-variable map
- Minimize 5-variable Boolean expressions using the Karnaugh map

A Karnaugh map for five variables (*ABCDE*) can be constructed using two 4-variable maps with which you are already familiar. Each map contains 16 cells with all combinations of variables *B*, *C*, *D*, and *E*. One map is for $A = 0$ and the other is for $A = 1$, as shown in Figure 4–41.

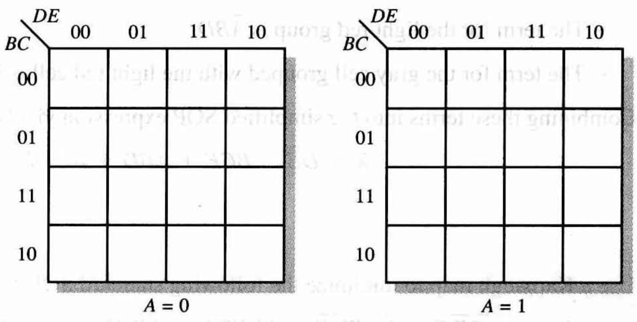


FIGURE 4–41
A 5-variable Karnaugh map.

## Cell Adjacencies

You already know how to determine adjacent cells within the 4-variable map. The best way to visualize cell adjacencies between the two 16-cell maps is to imagine that the $A = 0$ map is placed on top of the $A = 1$ map. Each cell in the $A = 0$ map is adjacent to the cell directly below it in the $A = 1$ map.

To illustrate, an example with four groups is shown in Figure 4–42 with the maps in a 3-dimensional arrangement. The 1s in the yellow cells form an 8-bit group (four in the $A = 0$

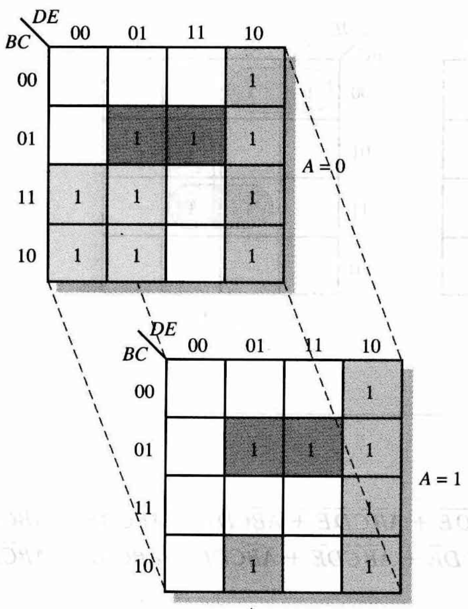


FIGURE 4–42
Illustration of groupings of 1s in adjacent cells of a 5-variable map.

map combined with four in the $A = 1$ map). The 1s in the orange cells form a 4-bit group. The 1s in the light red cells form a 4-bit group only in the $A = 0$ map. The 1 in the gray cell in the $A = 1$ map is grouped with the 1 in the lower right light red cell in the $A = 0$ map to form a 2-bit group.

***Determining the Boolean Expression*** The original SOP Boolean expression that is plotted on the Karnaugh map in Figure 4–42 contains seventeen 5-variable terms because there are seventeen 1s on the map. As you know, only the variables that do not change from uncomplemented to complemented or vice versa within a group remain in the expression for that group. The simplified expression taken from the map is developed as follows:

- The term for the yellow group is $D\overline{E}$.
- The term for the orange group is $\overline{B}CE$.
- The term for the light red group is $\overline{A}B\overline{D}$.
- The term for the gray cell grouped with the light red cell is $B\overline{C}\overline{D}E$.

Combining these terms into the simplified SOP expression yields

$$X = D\overline{E} + \overline{B}CE + \overline{A}B\overline{D} + B\overline{C}\overline{D}E$$

**EXAMPLE 4–33**

Use a Karnaugh map to minimize the following standard SOP 5-variable expression:

$$X = \overline{A}\overline{B}\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}C\overline{D}\overline{E} + \overline{A}BC\overline{D}\overline{E} + \overline{A}\overline{B}\overline{C}\overline{D}E + A\overline{B}\overline{C}\overline{D}\overline{E} + \overline{A}BC\overline{D}E$$
$$+ \overline{A}BCDE + A\overline{B}\overline{C}\overline{D}\overline{E} + A\overline{B}\overline{C}\overline{D}E + ABC\overline{D}E + ABCDE + A\overline{B}CDE$$

*Solution* Map the SOP expression. Figure 4–43 shows the groupings and their corresponding terms. Combining the terms yields the following minimized SOP expression:

$$X + \overline{A}\overline{D}\overline{E} + \overline{B}\overline{C}\overline{D} + BCE + ACDE$$

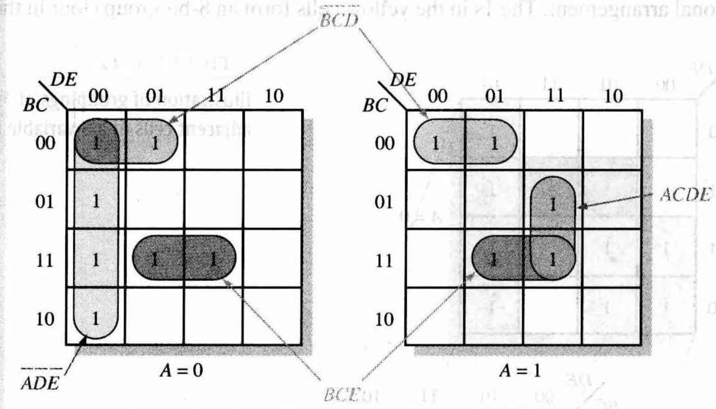


**FIGURE 4–43**

*Related Problem* Minimize the following expression:

$$Y = \overline{A}\overline{B}\overline{C}\overline{D}\overline{E} + \overline{A}\overline{B}C\overline{D}\overline{E} + \overline{A}B\overline{C}\overline{D}E + \overline{A}\overline{B}\overline{C}\overline{D}E + A\overline{B}\overline{C}\overline{D}\overline{E} + A\overline{B}C\overline{D}E + ABC\overline{D}\overline{E} + AB\overline{C}\overline{D}\overline{E}$$
$$+ \overline{A}\overline{B}\overline{C}D\overline{E} + \overline{A}\overline{B}CD\overline{E} + \overline{A}BCD\overline{E} + \overline{A}B\overline{C}D\overline{E} + A\overline{B}\overline{C}D\overline{E} + A\overline{B}CD\overline{E} + ABCD\overline{E} + AB\overline{C}D\overline{E}$$

**SECTION 4–10 CHECKUP**

1. **Why does a 5-variable Karnaugh map require 32 cells?**
2. **What is the expression represented by a 5-variable Karnaugh map in which each cell contains a 1?**

## 4–11 DESCRIBING LOGIC WITH AN HDL

Hardware description languages (HDLs) are tools for logic design entry, called text entry, that are used to implement logic designs in programmable logic devices. This section provides a brief introduction to VHDL and is not meant to teach the complete structure and syntax of the language. For more detailed information and instruction, refer to the footnote. Although VHDL provides multiple ways to describe a logic circuit, only the simplest and most direct programming examples of text entry are discussed here.

After completing this section, you should be able to

- State the essential elements of VHDL
- Write a simple VHDL program

The V in VHDL* stands for VHSIC (*V*ery *H*igh *S*peed *I*ntegrated *C*ircuit) and the HDL, of course, stands for hardware description language. As mentioned, **VHDL** is a standard language adopted by the IEEE (Institute of Electrical and Electronics Engineers) and is designated IEEE Std. 1076-1993. VHDL is a complex and comprehensive language and using it to its full potential involves a lot of effort and experience.

VHDL provides three basic approaches to describing a digital circuit using software: *behavioral, data flow,* and *structural.* We will restrict this discussion to the data flow approach in which you write Boolean-type statements to describe a logic circuit. Keep in mind that VHDL, as well as the other HDLs, is a tool for implementing digital designs and is, therefore, a means to an end and not an end in itself.

It is relatively easy to write programs to describe simple logic circuits in VHDL. The logical operators are the following VHDL keywords: **and, or, not, nand, nor, xor,** and **xnor.** The two essential elements in any VHDL program are the entity and the architecture, and they must be used together. The **entity** describes a given logic function in terms of its external inputs and outputs, called ports. The **architecture** describes the internal operation of the logic function.

In its simplest form, the entity element consists of three statements: The first statement assigns a name to a logic function; the second statement, called the *port* statement which is indented, specifies the inputs and outputs; and the third statement is the *end* statement. Although you would probably not write a VHDL program for a single gate, it is instructive to start with a simple example such as an AND gate. The VHDL entity declaration for a 2-input AND gate is

**Colons and semicolons must be used appropriately in all VHDL programs.**

```
entity AND_Gate2 is
  port (A, B: in bit; X: out bit);
end entity AND_Gate2;
```

The blue boldface terms are VHDL keywords; the other terms are identifiers that you assign; and the parentheses, colons, and semicolons are required VHDL syntax. As you can see, A and

*See Floyd, Thomas. 2003. *Digital Fundamentals with VHDL.* Prentice Hall; Pellerin, David and Taylor, Douglas. 1997. *VHDL Made Easy!* Prentice Hall; Bhasker, Jayaram. 1999. *A VHDL Primer,* 3 ed. Prentice Hall.