ENTITY example2 IS
    PORT ( x1, x2, x3, x4 : IN BIT ;
           f, g : OUT BIT ) ;
END example2 ;

ARCHITECTURE LogicFunc OF example2 IS
BEGIN
    f <= (x1 AND x3) OR (NOT x3 AND x2) ;
    g <= (NOT x3 OR x1) AND (NOT x3 OR x4) ;
END LogicFunc ;
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY fulladd IS
    PORT ( Cin, x, y : IN STD_LOGIC;
           s, Cout : OUT STD_LOGIC ) ;
END fulladd ;

ARCHITECTURE LogicFunc OF fulladd IS
BEGIN
    s <= x XOR y XOR Cin ;
    Cout <= (x AND y) OR (Cin AND x) OR (Cin AND y) ;
END LogicFunc ;

Figure 5.4 Full-adder

Figure 5.23 VHDL code for the full-adder
Figure 5.6 An $n$-bit ripple-carry adder

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY adder4 IS
  PORT ( Cin : IN STD_LOGIC;
         x3, x2, x1, x0 : IN STD_LOGIC;
         y3, y2, y1, y0 : IN STD_LOGIC;
         s3, s2, s1, s0 : OUT STD_LOGIC;
         Cout : OUT STD_LOGIC );
END adder4;

ARCHITECTURE Structure OF adder4 IS
  SIGNAL c1, c2, c3 : STD_LOGIC;
  COMPONENT fulladd
    PORT ( Cin, x, y : IN STD_LOGIC;
           s, Cout : OUT STD_LOGIC );
  END COMPONENT;
BEGIN
  stage0: fulladd PORT MAP ( Cin, x0, y0, s0, c1 );
  stage1: fulladd PORT MAP ( c1, x1, y1, s1, c2 );
  stage2: fulladd PORT MAP ( c2, x2, y2, s2, c3 );
  stage3: fulladd PORT MAP ( Cin => c3, Cout => Cout, x => x3, y => y3, s => s3 );
END Structure;

Figure 5.24 VHDL code for a four-bit adder
**Summary:**

**Entity Declaration:**

ENTITY entity_name IS
  PORT ( signal_name, signal_name, ... : signal_mode signal_type;
         signal_name, signal_name, ... : signal_mode signal_type;
         ...
         signal_name, signal_name, ... : signal_mode signal_type);
END entity_name;

**Signal types** discussed in book (p.687, etc.)
  - BIT
  - BIT_VECTOR
  - STD_LOGIC
  - STD_LOGIC_VECTOR
  - STD_ULOGIC
  - SIGNED
  - UNSIGNED
  - INTEGER
  - BOOLEAN
  - ENUMERATION.

**Signal modes:** (Table A2, p. 695)
- IN: input signal only
- OUT: output signal only; cannot be used on the right side of the simple signal assignment operator <=
- INOUT: a signal that is both an input and an output.
- BUFFER: an output signal. However, it can be used on the right side of the simple signal assignment operator <=

**Statement in the Architecture body** can be **structural** or **behavioral**.

**PORT MAP** statement is the only structural type statement in VHDL, used to instantiate a component that has been declared using a component declaration.

Simple signal assignment: signal_name <= expression;
Selected signal assignment: **WITH-SELECT-WHEN** statement
Conditional signal assignment: **WHEN-ELSE** statement

**PROCESS** statement (contains **sequential** statements)
  - Simple signal assignment statement <=
  - Variable assignment statement :=
  - IF-THEN-ELSE statement
  - CASE-WHEN statement
  - FOR-LOOP and WHILE-LOOP statements