DLD Lab

Introduction to VHDL

(Very High Speed Integrated Circuit Hardware Description Language)
Acknowledgment

This set of slides on VHDL are due to Brown and Vranesic.
ENTITY example1 IS
   PORT ( x1, x2, x3 : IN BIT;
          f         : OUT BIT);
END example1;

ARCHITECTURE LogicFunc OF example1 IS
BEGIN
   f <= (x1 AND x2) OR (NOT x2 AND x3);
END LogicFunc;

A simple logic function and corresponding VHDL code
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 ;

VHDL code for a four-input function
Logic circuit for four-input function
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux2to1 IS
    PORT ( w0, w1, s : IN STD_LOGIC;
           f : OUT STD_LOGIC )
END mux2to1;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
    WITH s SELECT
        f <= w0 WHEN '0',
        w1 WHEN OTHERS;
END Behavior;

Figure 6.27  VHDL code for a 2-to-1 multiplexer
Figure 6.1  A 2-to-1 multiplexer

(a) Graphical symbol
(b) Truth table
(c) Sum-of-products circuit
(d) Circuit with transmission gates
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;

ENTITY mux4to1 IS
    PORT ( w0, w1, w2, w3 : IN STD_LOGIC ;
          s         : IN STD_LOGIC_VECTOR(1 DOWNTO 0) ;
          f         : OUT STD_LOGIC ) ;
END mux4to1 ;

ARCHITECTURE Behavior OF mux4to1 IS
BEGIN
    WITH s SELECT
    f <= w0 WHEN "00",
        w1 WHEN "01",
        w2 WHEN "10",
        w3 WHEN OTHERS ;
END Behavior ;

Figure 6.28 VHDL code for a 4-to-1 multiplexer
Figure 6.2    A 4-to-1 multiplexer
LIBRARY ieee;
USE ieee.std_logic_1164.all;
PACKAGE mux4to1_package IS
  COMPONENT mux4to1
    PORT ( w0, w1, w2, w3 : IN STD_LOGIC ;
          s        : IN STD_LOGIC_VECTOR(1 DOWNTO 0) ;
          f        : OUT STD_LOGIC ) ;
  END COMPONENT ;
END mux4to1_package ;

Figure 6.28 Component declaration for the 4-to-1 multiplexer
Figure 6.3 Using 2-to-1 multiplexers to build a 4-to-1 multiplexer
LIBRARY ieee;
USE ieee.std_logic_1164.all;
LIBRARY work;
USE work.mux4to1_package.all;

ENTITY mux16to1 IS
    PORT ( w : IN STD_LOGIC_VECTOR(0 TO 15) ;
           s : IN STD_LOGIC_VECTOR(3 DOWNTO 0) ;
           f : OUT STD_LOGIC ) ;
END mux16to1 ;

ARCHITECTURE Structure OF mux16to1 IS
    SIGNAL m : STD_LOGIC_VECTOR(0 TO 3) ;
BEGIN
    Mux1: mux4to1 PORT MAP ( w(0), w(1), w(2), w(3), s(1 DOWNTO 0), m(0) ) ;
    Mux2: mux4to1 PORT MAP ( w(4), w(5), w(6), w(7), s(1 DOWNTO 0), m(1) ) ;
    Mux3: mux4to1 PORT MAP ( w(8), w(9), w(10), w(11), s(1 DOWNTO 0), m(2) ) ;
    Mux4: mux4to1 PORT MAP ( w(12), w(13), w(14), w(15), s(1 DOWNTO 0), m(3) ) ;
    Mux5: mux4to1 PORT MAP
           ( m(0), m(1), m(2), m(3), s(3 DOWNTO 2), f ) ;
END Structure ;

Figure 6.29 Hierarchical code for a 16-to-1 multiplexer
Figure 6.4  A 16-to-1 multiplexer
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY dec2to4 IS
  PORT ( w : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
         En : IN STD_LOGIC;
         y : OUT STD_LOGIC_VECTOR(0 TO 3) );
END dec2to4;

ARCHITECTURE Behavior OF dec2to4 IS
  SIGNAL Enw : STD_LOGIC_VECTOR(2 DOWNTO 0);
BEGIN
  Enw <= En & w;
  WITH Enw SELECT
    y <= "1000" WHEN "100",
         "0100" WHEN "101",
         "0010" WHEN "110",
         "0001" WHEN "111",
         "0000" WHEN OTHERS ;
END Behavior;

Figure 6.30 VHDL code for a 2-to-4 binary decoder
Figure 6.16  A 2-to-4 decoder
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux2to1 IS
    PORT ( w0, w1, s : IN STD_LOGIC;
            f : OUT STD_LOGIC );
END mux2to1;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
    f <= w0 WHEN s = '0' ELSE w1;
END Behavior;

Figure 6.31 A 2-to-1 multiplexer using a conditional signal assignment
ENTITY priority IS
    PORT ( w : IN STD_LOGIC_VECTOR(3 DOWNTO 0) ;
    y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0) ;
    z : OUT STD_LOGIC ) ;
END priority ;

ARCHITECTURE Behavior OF priority IS
BEGIN
    y <= "11" WHEN w(3) = '1' ELSE
         "10" WHEN w(2) = '1' ELSE
         "01" WHEN w(1) = '1' ELSE
         "00" ;
    z <= '0' WHEN w = "0000" ELSE '1' ;
END Behavior ;
<table>
<thead>
<tr>
<th>$w_3$</th>
<th>$w_2$</th>
<th>$w_1$</th>
<th>$w_0$</th>
<th>$y_1$</th>
<th>$y_0$</th>
<th>$z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>d</td>
<td>d</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>$x$</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$x$</td>
<td>$x$</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>$x$</td>
<td>$x$</td>
<td>$x$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 6.24  Truth table for a 4-to-2 priority encoder
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY priority IS
  PORT ( w : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
        y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
        z : OUT STD_LOGIC );
END priority;

ARCHITECTURE Behavior OF priority IS
BEGIN
  WITH w SELECT
    y <= "00" WHEN "0001",
         "01" WHEN "0010",
         "01" WHEN "0011",
         "10" WHEN "0100",
         "10" WHEN "0101",
         "10" WHEN "0110",
         "10" WHEN "0111",
         "11" WHEN OTHERS ;
  WITH w SELECT
    z <= '0' WHEN "0000",
         '1' WHEN OTHERS ;
END Behavior ;

Figure 6.33  Less efficient code for a priority encoder
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;

ENTITY compare IS
    PORT ( A, B : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
           AeqB, AgtB, AltB : OUT STD_LOGIC )
END compare;

ARCHITECTURE Behavior OF compare IS
BEGIN
    AeqB <= '1' WHEN A = B ELSE '0';
    AgtB <= '1' WHEN A > B ELSE '0';
    AltB <= '1' WHEN A < B ELSE '0';
END Behavior;

Figure 6.34 VHDL code for a four-bit comparator
Figure 6.26  A four-bit comparator circuit
LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;

ENTITY compare IS
  PORT ( A, B : IN SIGNED(3 DOWNTO 0) ;
        AeqB, AgtB, AltB : OUT STD_LOGIC ) ;
END compare ;

ARCHITECTURE Behavior OF compare IS
BEGIN
  AeqB <= '1' WHEN A = B ELSE '0' ;
  AgtB <= '1' WHEN A > B ELSE '0' ;
  AltB <= '1' WHEN A < B ELSE '0' ;
END Behavior ;

Figure 6.35  A four-bit comparator using signed numbers
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux2to1 IS
    PORT ( w0, w1, s : IN STD_LOGIC;
           f : OUT STD_LOGIC );
END mux2to1;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
    PROCESS ( w0, w1, s )
    BEGIN
        IF s = '0' THEN
            f <= w0;
        ELSE
            f <= w1;
        END IF;
    END PROCESS;
END Behavior;

Figure 6.38 A 2-to-1 multiplexer specified using an if-then-else statement
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux2to1 IS
    PORT ( w0, w1, s : IN STD_LOGIC;
           f : OUT STD_LOGIC )
END mux2to1;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
    PROCESS ( w0, w1, s )
    BEGIN
        f <= w0;
        IF s = '1' THEN
            f <= w1;
        END IF;
    END PROCESS;
END Behavior;

Figure 6.39  Alternative code for a 2-to-1 multiplexer
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY priority IS
    PORT ( w : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
          y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
          z : OUT STD_LOGIC );
END priority ;

ARCHITECTURE Behavior OF priority IS
BEGIN
    PROCESS ( w )
    BEGIN
        IF w(3) = '1' THEN
            y <= "11" ;
        ELSIF w(2) = '1' THEN
            y <= "10" ;
        ELSIF w(1) = '1' THEN
            y <= "01" ;
        ELSE
            y <= "00" ;
        END IF ;
    END PROCESS ;
z <= '0' WHEN w = "0000" ELSE '1' ;
END Behavior ;

Figure 6.40  A priority encoder specified using if-then-else
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY priority IS
  PORT ( w : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
         y : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
         z : OUT STD_LOGIC );
END priority;

ARCHITECTURE Behavior OF priority IS
BEGIN
  PROCESS ( w )
  BEGIN
    y <= "00" ;
    IF w(1) = '1' THEN y <= "01" ; END IF ;
    IF w(2) = '1' THEN y <= "10" ; END IF ;
    IF w(3) = '1' THEN y <= "11" ; END IF ;
    z <= '1' ;
    IF w = "0000" THEN z <= '0' ; END IF ;
  END PROCESS ;
END Behavior ;

Figure 6.41  Alternative code for the priority encoder
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY compare1 IS
  PORT ( A, B : IN STD_LOGIC;
         AeqB : OUT STD_LOGIC );
END compare1;

ARCHITECTURE Behavior OF compare1 IS
BEGIN
  PROCESS ( A, B )
  BEGIN
    AeqB <= '0';
    IF A = B THEN
      AeqB <= '1';
    END IF;
  END PROCESS;
END Behavior;

Figure 6.42 Code for a one-bit equality comparator
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux2to1 IS
    PORT ( w0, w1, s : IN STD_LOGIC;
          f      : OUT STD_LOGIC );
END mux2to1;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
    PROCESS ( w0, w1, s )
    BEGIN
        CASE s IS
            WHEN '0' =>
                f <= w0 ;
            WHEN OTHERS =>
                f <= w1 ;
        END CASE ;
    END PROCESS ;
END Behavior ;

Figure 6.45  A CASE statement that represents a 2-to-1 multiplexer
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY dec2to4 IS
    PORT ( w : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
           En : IN STD_LOGIC;
           y : OUT STD_LOGIC_VECTOR(0 TO 3) );
END dec2to4 ;
ARCHITECTURE Behavior OF dec2to4 IS
BEGIN
    PROCESS ( w, En )
    BEGIN
        IF En = '1' THEN
            CASE w IS
            WHEN "00" => y <= "1000";
            WHEN "01" => y <= "0100";
            WHEN "10" => y <= "0010";
            WHEN OTHERS => y <= "0001";
            END CASE;
        ELSE
            y <= "0000";
        END IF;
    END PROCESS;
END Behavior ;

Figure 6.46 A 2-to-4 binary decoder
LIBRARY ieee;
USE ieee.std_logic_1164.all;
ENTITY seg7 IS
    PORT ( bcd : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
           leds : OUT STD_LOGIC_VECTOR(1 TO 7) );
END seg7;
ARCHITECTURE Behavior OF seg7 IS
BEGIN
    PROCESS ( bcd )
    BEGIN
        CASE bcd IS
            WHEN "0000"  => leds <= "1111110" ;
            WHEN "0001"  => leds <= "0110000" ;
            WHEN "0010"  => leds <= "1101101" ;
            WHEN "0011"  => leds <= "1111001" ;
            WHEN "0100"  => leds <= "0110011" ;
            WHEN "0101"  => leds <= "1011011" ;
            WHEN "0110"  => leds <= "1011111" ;
            WHEN "0111"  => leds <= "1110000" ;
            WHEN "1000"  => leds <= "1111111" ;
            WHEN "1001"  => leds <= "1110011" ;
            WHEN "1010"  => leds <= "1011011" ;
            WHEN "1011"  => leds <= "1110000" ;
            WHEN "1100"  => leds <= "1111111" ;
            WHEN "1101"  => leds <= "1110011" ;
            WHEN OTHERS => leds <= "-------" ;
        END CASE ;
    END PROCESS ;
END Behavior ;

Figure 6.47  A BCD-to-7-segment decoder