VHDL Precedence

**Example**

\[ x_1 \text{ AND } x_2 \text{ OR } x_3 \text{ AND } x_4 \]

does not mean \[ x_1 x_2 + x_3 x_4 \]

Because \text{AND} does not have precedence over \text{OR}. To have the desired meaning, it must be written as

\[ (x_1 \text{ AND } x_2) \text{ OR } (x_3 \text{ AND } x_4) \]

---

**VHDL Review – Case Statements**

Case statements: Executes one of several sequences of statements, based on the value of a single expression. The syntax is as follows:

\[
\text{case expression is}
\begin{align*}
\text{when choices } & \rightarrow \text{ sequential statements} \\
\text{when choices } & \rightarrow \text{ sequential statements} \\
\text{when others } & \rightarrow \text{ sequential statements}
\end{align*}
\]

---

**Case Example – Selecting a grade**

```vhdl
library ieee;
use ieee.std_logic_1164.all;

entity GRD_201 is
    port (
        VALUE: in integer range 0 to 100;
        A, B, C, D: out bit)
end entity;

architecture behav_grd of GRD_201 is
begin
    process (VALUE)
    begin
        case VALUE is
            when 51 to 60 => D <= '1';
            when 61 to 70 | 71 to 75 => C <= '1';
            when 76 to 85 => B <= '1';
            when 86 to 100 => A <= '1';
            when others => F <= '1';
        end case;
    end process;
end architecture;
```

---

**Case Example – 4 to 1 MUX**

```vhdl
entity MUX_4_1 is
    port ( SEL: in std_logic_vector(2 downto 1); A, B, C, D: in std_logic; Z: out std_logic);
end entity;

architecture behav_MUX41 of MUX_4_1 is
begin
    process (SEL, A, B, C, D)
    begin
        case SEL is
            when "00" => Z <= A;
            when "01" => Z <= B;
            when "11" => Z <= D;
            when others => Z <= 'X';
        end case;
    end process;
end architecture;
```

---

**VHDL Review - Component**

Component: a sub-circuit of a VHDL file, that either must be a previously defined entity in a library or entity within the VHDL file. Syntax for calling the component:

```vhdl
component component_name is
    port (port_signal_names: mode type);
end component [component_name];
begin
    port map [list of actual signals]
```
Example – 4 Bit Adder

-- Example of a four bit adder
library ieee;
use ieee.std_logic_1164.all;

-- definition of a full adder
entity FULLADDER is
port(a, b,c: in std_logic;
sum, carry: out std_logic);
end FULLADDER;

architecture fulladder_behav of FULLADDER is
begin
sum <= (a xor b) xor c;
carry<=(a and b) or (c and (a xor b));
end fulladder_behav;

-- 4-bit adder
library ieee;
use ieee.std_logic_1164.all;

entity FOURBITADD is
port(a,b: in std_logic_vector(3 downto 0);
Cin: in std_logic;
sum: out std_logic_vector(3 downto 0);
Cout, V: out std_logic);
end FOURBITADD;

architecture fouradder_structure of FOURBITADD is
signal c: std_logic_vector (4 downto 0);
component FULLADDER port
(a, b, c: in std_logic;
sum, carry: out std_logic);
end component;

begin
FA0: FULLADDER port map(a(0), b(0), Cin, sum(0), c(1));
FA1: FULLADDER port map(a(1), b(1), C(1), sum(1), c(2));
FA2: FULLADDER port map(a(2), b(2), C(2), sum(2), c(3));
FA3: FULLADDER port map(a(3), b(3), C(3), sum(3), c(4));
V <= c(3) xor c(4);
Cout<= c(4);
end fouradder_structure;
end FOURBITADD;

Lab #2

• Due Thursday 9/18/14
• Hardcopy of team lab report
• Total of 6 parts
• Knowledge gained from previous parts is utilized in later parts
• Include a copy of your VHDL code for each part
• Document and describe what was learned
• Combines Lab1 with the use of 7-segment display
• A total of 6 are available on DE2 board
• Can represent digits 0-9 and hex letters A-F

7-Segement Display Pin out

Note: Applying a low logic level to a segment will light it up and applying a high logic level turns it off.

References

• VHDL Primer:
http://www.seas.upenn.edu/~ese171/vhdl/vhdl_primer.html