ECEN 4856

Lecture 4: Lab 2 Discussion

VHDL Precedence

Table A.1 The VHDL operators.

	Operator Class	Operator
Highest precedence	Miscellaneous	**, ABS, NOT
	Multiplying	*, /, MOD, REM
	Sign	+, –
	Adding	+, –, &
	Relational	=, /=, <, <=, >, >=
Lowest precedence	Logical	AND, OR, NAND, NOR, XOR, XNOR

Example,

x1 AND x2 OR x3 AND x4

does not have mean

x1x2 + x3x4

Because **AND** does not have precedence over **OR**. To have the desired meaning, it must be written as

(x1 AND x2) OR (x3 AND x4)

VHDL Review – Case Statements

(Page 358 in textbook)

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

```
case expression is
      when choices =>
            sequential statements
      when choices =>
            sequential statements
            -- branches are allowed
      [when others => sequential statements ]
end case;
```

Case Example – Selecting a grade

```
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 GRD_201;
architecture behav_grd of GRD_201 is
begin
     process (VALUE)
       A <= '0';
       B <= '0';
       C <= '0';
       D <= '0';
       F <= '0';
       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 behav_grd;

The "|" symbol used to show how multiple values can be on a line.
```

Case Example – 4 to 1 MUX

```
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 MUX_4_1;
architecture behav_MUX41 of MUX_4_1 is
begin
   PR_MUX: process (SEL, A, B, C, D)
   begin
     case SEL is
          when "00" => Z <= A;
          when "01" => Z <= B;
          when "10" => Z <= C;
          when "11" => Z <= D;
          when others => Z <= 'X';
     end case;
   end process PR_MUX;
end behav_MUX41;
```

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:

component *component_name* ***is***
  port (*port_signal_names*: **mode** *type*;
  port_signal_names: **mode** *type*;
  :
  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;

(Continue on the right column)
```

```
-- 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 fouradder_structure;
```

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

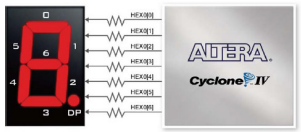

Figure 4-10 Connections between the 7-segment display HEX0 and Cyclone IV E FPGA

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
- *Fundamentals of Digital Logic with VHDL Design:* Stephen Brown and Zvonko Vranesic ISBN 978–0–07–352953–0