Comp 331 – Intro to Computer Hardware

7-Segment Display

1. 7-Segment Decoder
   • Karnaugh Maps to Logic Statements
   • Case Statement
2. Example 13: hex7seg.vhd (case stmt)
3. Example 14: Multiplexing hex7seg.vhd
4. Example 15: Clock Multiplexing hex7seg.vhd

Project: Example 13

Do the following

1. Create/Compile hex7seg.vhd (case stmt version)
   • use Listing 5.17 or 5.18
2. Create/Compile hex7seg_top
   • use listing 5.19
3. Synthesize & Implement on Basys-2 Board
4. Test digits 0 – F (instructor check)
5. Print out both .vhd files (one document)

Example 14

Multiplexing the four 7-segment displays

Recall how to insert a separate component in the architecture section of a .vhd file.

Note that Aldec-Active-HDL supports a "copy and paste" method for this

architecture hex7seg_top of hex7seg_top is

   component hex7seg is
   port ( ...);
   end component;

   begin
   ...;
   C1: hex7seg port map ...
   ...;
end hex7seg_top

Example 15

Using the Clock to Multiplex 30x/sec

Instructor Check!
• What happens if two buttons are pushed simultaneously? Explain
5. Print out mux7seg_top.vhd
Details of “Clock Divider”: clk2bit

clkdiv is a 20 bit vector which is incremented on each rising clock (clk) signal. \( s = \text{clkdiv}[19:18] \) is a 4 phase “signal” (00, 01, 10, 11 ...) obtained from clkdiv. clr (clear) is a signal to reset clkdiv to 0 if asserted.

\[
\text{process}(
\text{clk, clr})
\text{begin}
\text{if} \ clr = '1'
\text{then}
\text{clkdiv} <= ('0');
\text{elsif} \ \text{clk}'\text{event and clk} = '1'
\text{then}
\text{clkdiv} <= \text{clkdiv} + 1;
\text{endif};
\text{end process};
\]

A 50 MHz clk will “drive” clkdiv from all 0’s to all 1’s in about a one fiftieth of a second – but since \( s = \text{clkdiv}[19:18] \) the 4 phase repeating “signal” 00, 01, 11, ... will change every hundredth of a second

Details of Anode Select

Based on the value of 2-bit vector \( s \) (00, 01, 10, or 11), anode select sets the corresponding bit in the “an” vector to 0

\[
\text{signal aen: STD_LOGIC_VECTOR(3 downto 0);}\]
\[
\text{signal an: "1111";}\]
\[
\text{process}(s, aen)
\text{begin}
\text{an} <= "1111";
\text{if} \ aen(\text{conv_integer}(s)) = '1'
\text{then}
\text{an}(\text{conv_integer}(s)) <= '0';
\text{end if};
\text{end process};
\]

Example 15A: x7seg

Do the following
1. Again copy hex7seg.vhd to your project
2. Use Listing 5.21 for x7seg.vhd
   • Insert declaration and implementation of hex7seg.vhd in place of process code found in listing
3. Use Listing 5.22 for x7seg_top.vhd
4. Instructor Check!
5. Print out in one listing copies of all .vhd files.

Example 15B: x7segb

7-segment display with leading blanks
Do the following
1. Again copy hex7seg.vhd to your project
2. Use Listing 5.23 for x7segb.vhd
   • Only difference is how \( aen \) vector is initialized
   • Again insert declaration and implementation of hex7seg.vhd in place of process code found in listing
3. Use Listing 5.24 for x7segb_top.vhd
   • Note that input is 8 switches plus 3 buttons concatenated to 01010.
4. Instructor Check!
5. Print out in one listing copies of all .vhd files.