Enhanced Priority Encoder

Signature and Grading Sheet

Group #:_____ Name(s):_____ .

Signature

Section 4.2(f): _____.

Grading

- Section 4.1(a): Top-level diagram (10 points):_____.
  Attach a page.
- Section 4.1(b): VHDL code (25 points):_____.
  Attach code printout (with proper header and comment)
- Section 4.1(c) Simulation (15 points):_____.
  Attach simulation timing diagram printout.
- Section 4.1(d) (5 points):_____.
  Attach one-page report printout with # of LEs circled.
- Section 4.2(a): VHDL code (10 points):_____.
  Attach code printout (with proper header and comment)
- Section 4.2(f) demonstration (15 points):_____.
- Section 4.3(a)(b)(c)(d): (20 points):_____.
  4.3(a): Attach the row-generation code.
  4.3(b): Attach only first two pages of VHDL code.
  4.3(c): Attach simulation timing diagram printout.
  4.3(d): Attach one-page report printout with # of LEs circled.

Total points: _____.

Experiment
Enhanced Priority Encoder

1 Purpose

Use VHDL routing constructs to design a special priority encoder.

2 Reading

Chapter 3 of *Embedded SoPC Design with Nios II Processor and VHDL Examples*

3 Project specification

An enhanced-priority encoder returns the codes of the highest and second-highest priority requests. The input is the 10-bit r signal, in which the r(9) has the highest priority and r(0) has the lowest priority. The outputs are fst and snd, which are the 4-bit binary codes of the highest and second-highest priority requests, respectively. The input and output signals are

- input:
  - r: 10-bit input request
- output
  - fst: 4-bit output, the binary code of the highest priority request
  - snd: 4-bit output, the binary code of the second-highest priority request

An output becomes "1111" when there is no active request.

4 Design Procedures

4.1 Encoder circuit

The best way to derive efficient VHDL code is to think in terms of hardware. First, draw a conceptual top-level schematic diagram (i.e., what you will do if no HDL/synthesis software is available) and then derive the code accordingly. The score is based on not only the correctness of the code but also the quality and size of the design and circuit.

(a) Derive a conceptual top-level diagram. The diagram should contain 3 to 10 blocks, each to be realized by a VHDL statement.

(b) Derive VHDL code according to conceptual top-level diagram. The entity declaration of this design is:

```
entity enhanced_prio is
   port(
      r: in std_logic_vector(9 downto 0);
      fst, snd: out std_logic_vector(3 downto 0)
    );
end enhanced _prio;
```

Proper header and comments should be included. In the comments, indicate the correspondence between VHDL statements and blocks. No VHDL process can be used.

(c) Compile the design and perform simulation. The simulation should include at 10 test patterns to verify various input conditions. In the timing diagram, the input and output signals should be clearly arranged and represented in proper format so that the simulation result can be easily understood.

(d) Look at the compiling report and determine the number of LEs used in circuit.

4.2 Testing circuit with 7-segment display

Use 10-bit switch as the request signal. Decode the fst and snd signals and display the two codes in hex format in two seven-segment LED displays. An LED displays 0 to 9 patterns if there is an active request and displays "L" (for null) if there is no active request.

(a) Derive VHDL code to include the additional feature.

(b) Perform the pin assignment. If you wish, you can use another top-level "wrapping" circuit.

(c) Synthesize the circuit.

(d) Download the configuration file to DE1 board.

(e) Use the switches and seven-segment LED displays to verify the operation of physical circuit.

(f) Demonstrate the circuit to instructor and get signature.

4.3 Implementation with truth table

A combination circuit can be exhaustively specified by a truth table, which can be realized by one VHDL selected assignment statement. For the 10-request enhanced-priority encoder in this experiment, a 2^{10}-by-8 table is needed. The architecture body looks like

```
architecture table_arch of enhanced_prio is
   signal both: std_logic_vector(7 downto 0); -- codes of 1st & 2nd
begin
   with r select
      both <=
         "11111111" when "0000000000",  -- 1st:null; 2nd:null
         "00001111" when "0000000001",  -- 1st:0; 2nd:null
         "00011111" when "0000000010",  -- 1st:1; 2nd:null
         "00010000" when "0000000011",  -- 1st:1; 2nd:0
         ...                            -- 1024 lines

   fst <= both(7 downto 4);
   snd <= both(3 downto 0);
end table_arch;
```

While the code is straightforward, manually completing the 1024 rows is tedious and time consuming. If you wish, you can write a program in the language of your choice (C, Java, Perl, etc.) to generate theses rows and copy/paste them to the VHDL code.

(a) Write a code generation program if you wish.

(b) Complete the VHDL code.

(c) Compile the design and perform simulation. The simulation should include at 10 test patterns to verify various input conditions. In the timing diagram, the input and output signals should be clearly arranged and represented in proper format so that the simulation result can be easily understood.

(d) Look at the compiling report and determine the number of LEs used in circuit. Compare it with that of Section 4.1(d).