CMPE 415 – Homework/Project #6

Spring 2015

Work individually, but I recommend working with someone in the class nearby so you can help each other when you get stuck, with consideration to the course collaboration policy (please read it in the course website). Please send me email if something isn't clear and I will update the assignment. Changes are logged at the bottom of this page.

Before getting started, you should go through the verilog notes located under Course Readings on the course home page.

A paper copy of everything and electronic copies of all your code and testing files (all in one zipped file) are due at the beginning of class on the due date.

Notes:

- [15% of points] Clearly state whether your design is fully functional, and state the failing sections if any exist.
- Make sure your design and code are easily readable and understandable (clear and well commented).
  - Up to 5% extra credit will be given for especially thorough, well-documented, or insightful solutions.
- *** Where three '*'s appear in the homework, perform the required test(s) and turn in a printout of either:
  1. a table printed by your verilog testbench module listing all inputs and corresponding outputs,
  2. an Isim waveform plot which clearly shows (labeled and highlighted) corresponding inputs and outputs, or
  3. a section of testing code which clearly compares the designed circuit and a simple reference circuit, and two short cut & paste sections of text from your simulation (one for pass, and one for fail where you purposely make a slight change to your reference code to make it fail) that look something like this:
     Error: input=0101, out_module=11110000, out_ref=11110001

In all three options, each test case must be marked whether the output is correct or not.

Keep "hardware" modules separate from testing code. Instantiate a copy of your processing module(s) in your testing module (the highest level module) and drive the inputs and check the outputs from there.
Problem 1

Using the Hamming coding described in the class, determine whether there is an error in each of the following ECC words (you need to write the equations and explain in detail), and if so, determine the corrected ECC word and the original data value.

a) 100100011010
b) 000110111000
c) 111011011101

Problem 2

Using Hamming code described in class, design an error correction code (ECC) for a 4-bit data word.

Reminder:
- Required number of check bits is \(\log_2 N + 1\), where \(N\) is data word length
- ECC bits whose indices are powers of two are used as check bits.
- If we write the indices of ECC bits in binary, the check bit with a 1 in position \(i\) of its index is the XOR of data ECC bits that have a one in position \(i\) of their indices.

[5 pts] What is the length of total code word? Which bits are check bits and which one are data bits? Compute the ECC bits for 4-bit data 0110 and write the complete code word.

[20 pts] Write the verilog for the module that checks if there is an error in the received data. The verilog module has the received word as input and has two outputs error and error-bit. If there is any error, then the verilog module must locate the error bit and make the error signal to be 1 and send out the location of error-bit. Otherwise the error signal remains zero.

**Verified Behavioral Simulation**

- [20 pt] Write a testbench that can test the block with the input values in Problem 1.
Implementation

- [5 pt] Perform Synthesis and Place and Route and report total slice count, and other FPGA resource counts that are listed in Summary Report.

- [5pt] Report design speed using Place and Route Static Timing Analysis (under Place and Route). Slack must be a minimum positive number (near zero).

- [5 pt] Using the XPower analyzer tool (under Place and Route), report the power dissipation of the design for your clock specification that design can operate.

Written Report

- Submit all files with a complete report explaining the details of the design and block diagrams. If you need to pipeline the design you need to clearly state and show in the block diagrams.