Midterm Examination
CSE 310
CSU San Bernardino
Winter 2016
Dr. Yasha Karant

This examination is open notes, open book. **The examination is due no later than midnight Tuesday 15 February 2016.**

Point values are as given; there are 100 total points possible. ***Please read each question carefully.*** Display all references. For full credit, **all work must be shown**; this includes calculations, derivations, proofs, graphs, as well as your arguments and reasoning. Your argumentation and reasoning is important; results without detailed arguments will not result in much credit. **All work must be single authorship, and the University policy on plagarism will be enforced.**

You are required to submit your work as a single PDF file using the submission mechanism of Blackboard. This statement means one file, not many, and not in JPEG, MS Word DOC, or any format other than PDF. Make certain that the file name is unique (e.g., your-name-cs310midterm.pdf, not just midterm.pdf or other largely non-unique name). Note that Apache OpenOffice, LibreOffice, or TeXstudio will output your work as a PDF file and each is licensed for free for all major platforms and environments. If you must handwrite your work or any parts thereof, you can scan your work and convert the output to PDF. There exist free use PDF conversion sites on the Web in a worst case.

Problems

1. **[50 points]** We have discussed gray code as well as simple arithmetic circuits.

   1.1. Define gray code and explain why it is used (hint: consider a set of physical toggle switches and a human operator controlling these switches).

   1.2. Consider a 5 bit gray code.

      1.2.1. How many entities can 5 bits encode? – explain your reasoning, do not just state an answer.

      1.2.2. Assume the ordinary binary representation of a hex digit (e.g., 0xA is binary 1010). Write a 5 digit gray code sequence from 0 to the largest number that can be encoded in five bits. Then write both the ordinary binary and ordinary hexadecimal representation corresponding to each number in that sequence.

   1.3. Consider ordinary unsigned addition for a 3 bit gray code values – i.e., the sum of two 3 bit gray code values as a gray code value.

      1.3.1. Given all possible 3 bit input values for a 3 bit gray code, what are the possible output values as gray codes? How are these gray codes interpreted as "ordinary" numbers?

      1.3.2. If you are constructing a full adder, there are two additional values: carry in and carry out. Extend your solution to 1.3.1. include these two additional values, and display the result as a truth table using 1 and 0 entries.

1.3.3. Using the truth table in 1.3.2., construct the Karnaugh map for the 3 bit gray code full adder. Does this Karnaugh map reveal any hazards?

1.3.4. Write an appropriate Verilog program to implement the 3 bit gray code adder.

1.3.5. Combine appropriate 3 bit gray code full adders to produce a single 6 bit gray code full adder.

1.4. Consider an anti-gray code which maximizes hamming distance for adjacent values in a 4-bit binary sequence, starting at 0.
From : https : //en.wikipedia.org/wiki/Hamming_distance
In information theory, the Hamming distance between two strings of equal length is the number of positions at which the corresponding symbols are different. In another way, it measures the minimum number of substitutions required to change one string into the other, or the minimum number of errors that could have transformed one string into the other.

1.4.1. Design and write such a sequence in binary representation.

1.4.2. What is the average hamming distance for the values in this sequence?

1.4.3. What real-world applications would such a sequence have?

2. **[50 points]** We have discussed 1-hot code.

2.1. Define 1-hot code and explain why it is used (hint: consider a set of physical toggle switches and a human operator controlling these switches).

2.2. Display the 3 bit regular binary and the 1-hot code that represents the same value.

2.3. 3 bits correspond to what radix?

2.4. A 1-cold code is the same as a 1-hot except that the 1-cold has a low where the 1-hot has a high, and vice versa. How does one convert a 1-hot code to a 1-cold code?

2.5. Write a truth table that shows the conversion of 3 bit ordinary binary to a 1-cold code

2.6. Write a Verilog program that converts a 3 bit binary number into 1-cold code.

2.7. Are there any hazards in your circuit? If so, how would you correct these?