Problem #1

(a) A digital system is required to amplify a binary-encoded audio signal. The user should be able to control the signal amplitude from \textbf{minimum to maximum in 100 increments}. What is the \textbf{minimum number of binary bits} required to encode the user-specified amplitude?

(b) Excess-3 code (Katz page 499) is a variation of binary-coded decimal (BCD) code. Each decimal digit is represented by a 4-bit code that is \textbf{three more than the associated BCD code}. For example, 0 is encoded in excess-3 as 0011b, 1 is encoded in excess-3 as 0100b, etc. Design a \textbf{single-output combinational logic circuit} that \textbf{outputs a 1 when the input to the circuit in 4-bit excess-3 code is a prime number}. For all other (non-prime) \textbf{legal 4-bit excess-3 numbers} applied to the inputs, the output is a 0. \textbf{Assume complement inputs are available} and implement the circuit using:

(i) One 16-input, four control-line multiplexer only
(ii) One 8-input, three control-line multiplexer only.
(iii) One 4-input, two control-line multiplexer and a minimum number of simple logic gates (INV, NAND, NOR, ND, OR, XNOR)

Problem #2

(a) What is the better way to implement arithmetic in a binary computer: \textbf{one's complement or two's complement}? Why? \textbf{Include all of the arguments you can think of for and against your answer}.

(b) Does the state-machine opposite have any \textbf{equivalent states}? If so, \textbf{which states are equivalent}? Show all work.

(c) Design a \textbf{4-bit ripple up-counter} using \textbf{positive edge-triggered D flip flops} and a minimum number of combinational logic gates. Show the schematic diagram.
Problem #3

Design a Moore machine that detects the number 9 encoded in binary (1001b) and in excess-3 (1100b). The machine should reset after each detection (i.e. overlapping sequences are ignored). Sample input (X) and output (Z) sequences are given below:

X = 1001100011001...
Z = 0001000000010...

(a) Draw a state transition diagram for the Moore machine.
(b) Use an implication table to determine if any states are equivalent. If so, list the equivalent states and then redraw your (now minimized) state diagram.
(c) Using D flip-flops and the state-assignment rules discussed in class, indicate all adjacency constraints for an optimal state encoding and determine an optimal encoding, listing the state codes for each state in the machine.
(d) Write equations for the next-state logic only (not the output logic) using a minimal NOR-NOR two-level representation.

Posted by HKN (Electrical Engineering and Computer Science Honor Society)
University of California at Berkeley
If you have any questions about these online exams please contact mailto:examfile@hkn.eecs.berkeley.edu