DIGITAL DESIGN PROBLEM

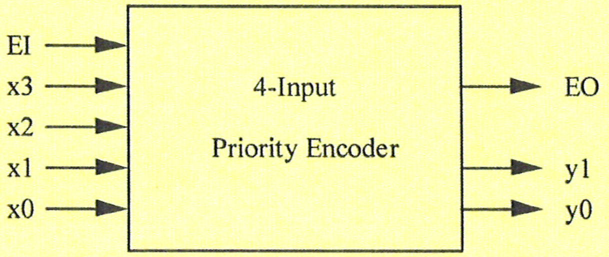
Problem 1 [0.5 points]
Implement a magnitude comparator circuit which compares two-bit binary numbers $A=A_1A_0$ and $B=B_1B_0$, and provides 3 outputs ($A>B$, $A=B$, $A<B$), using three 8-to-1 multiplexers and no additional gates.

Problem 2 [1.5 points]

A 4-input priority encoder with enable is shown below. When EI is 0, all outputs are 0. When EI is 1,

i) and all 4 xi inputs are 0, EO is 1 and y1=y0=0

ii) and not all 4 xi inputs are 0, EO is 0 and y1 and y0 are the binary encoding for the highest priority xi input that is 1 (For example, x3 is highest priority whose binary encoding is 11. x1 is lowest priority with binary encoding 01 and so on).



(a) [0.4 points] Draw a two-level NAND-gate implementation of this priority encoder. Assume both true and complementary inputs are available.

(b) [0.4 points] Draw a two-level NOR-gate implementation of this priority encoder. Assume both true and complementary inputs are available.

(c) [0.7 points] Draw a diagram of an 8-input priority encoder of the same type using two 4-input priority encoders and a few logic gates.

Problem 3 [2.0 points]

(a) [0.5 points] A basic lawn sprinkler system waters the lawn at 8am, once every 3 days. Its operation can be modeled as a Moore-style finite state machine, where a clock period is one day, and a binary output W indicates whether the sprinkler is activated (W=1) or not (W=0) on that day. Show the transition diagram for this sprinkler control system.

(b) [0.5 points] An advanced sprinkler system has a rain sensor. The presence of rain during a particular day is indicated by the binary input R (with values 0/1 indicating No/Yes_rain). This sprinkler waters the lawn once every 3 days, assuming there has been no rain during previous 2 days. Show the transition diagram (Moore-style) for this advanced sprinkler system.

(c) [1.0 points] Implement the transition diagram of a sprinkler system in part (b) using a 4-bit counter with parallel load capability and external gates. The counter is specified by its operation table shown below. Use the most appropriate state assignment for this counter implementation. (*Implementation* means showing the block diagram with clearly labeled inputs/outputs, and derivation of the Boolean equations for control/data inputs of the counter).

Counter operation table:
(All functions are synchronous with a clock)

Clear	*Load*	*Count*	*Function*
0	X	X	Clear
1	0	0	No change
1	1	X	Load
1	0	1	Count up