Roll no:

CS225: Final Examination (Spring 2018)

Max: 100 points

Q1: Suppose we have a 7-bit computer that uses IEEE floating-point arithmetic where a floating point number has 1 sign bit, 3 exponent bits, and 3 fraction bits. For each of the following, write the binary value and the corresponding decimal value of the 7-bit floating point number that is the closest available representation of the requested number. If rounding is necessary use round-to-nearest. Give the decimal values either as whole numbers or fractions. The first line is filled in for you.

Number	Binary	Decimal
0	0 0000 000	0.0
-0.125		
-3.1		
12.25		
Smallest positive normalized number		
Largest positive number $< \infty$		
Smallest positive de-normalized number > 0		
Largest de-normalized Number		
5/8		
1.0		
5.0625		

(10)

(b) The associative law for addition says that $a + (b + c) = (a + b) + c$. This holds for regular arithmetic, but does not always hold for floating-point numbers. Using the 7-bit floating-point system described above, give an example of three floating-point numbers a, b, and c for which the associative law does not hold, and show why the law does not hold for those three numbers.

(6)

Q2: Consider a combination logic sub system that performs a 2-bit addition function. It has two 2-bit inputs AB and CD and forms the 3-bit sum, XYZ.

a) Draw the truth tables for X,Y and Z

b) Minimize the function using 4-variable K-maps to derive minimized sum-of-products form.

c) What is the relative performance to compute the resulting sum bits of the 2-bit adder compare to two full adders connected together?

(4+6+4)

Q3: Simplify the following

a) $f(a,b,c,d,e) = \Pi M(0,4,18,19,22,23,25,29)$

b) $f(a,b,c,d,e) = \Sigma m(0,4,18,19,22,23,25,29)$

(5)

Q4: Simplify

$f(a,b,c,d,e,f) = \Sigma m(3,7,12,14,15,19,23,27,28,29,31,35,39,44,45,46,48,49,50,52,53, 55,56,57,59)$

(5)

Q5: A finite state machine has one input and one output. The output becomes 1 and remains 1 thereafter when at least two 0's and two 1's have occurred as inputs, regardless of the order of appearance. Draw a state transition diagram for the machine. Show the block schematic of the implementation with clearly marking inputs, state registers and output (logic level implementation not required)

(7+3)

Q6: Design a 4 bit register with two control inputs s1 and s0, 4 data inputs I3, I2,I1, and I0 and 4 data outputs Q3, Q2, Q1, and Q0.
When s1s0=00, the register maintains its value.
When s1s0=01, the register loads $I_3I_2I_1I_0$
When s1s0=10, the register loads 1010
When s1s0=11, the register complements itself, so for example 0000 would become 1111, and 1010 would become 0101.

(5)

Q7: (a) Consider a dynamic RAM that must be given a refresh cycle 64 times per ms. Each refresh operation requires 150ns; a memory cycle requires 250ns. What percentage of the memory's total operating time must be given to refreshes?
(b) A computer system has 16-bit wide data bus uses RAM chips of 64K x 8. How many chips are needed and how should their address lines be connected to provide a memory capacity of 1M

words . Show the address mapping for each?

(7+3)

Q8: Explain how to configure a CLB (configurable logic block) to perform the following functions. Assume 4-bit/3bit look-up tables as discussed in the class. Show the truth tables with clear input variables and mapping(block schematic assuming 4-bit/3bit CLBs).

(a) F=a'b'c+ abc' and G= ab'
(b) Y=jklmpqr

(5+5)

Q9: Let f be completely specified Boolean functions. Assume that x be a support variable and $f_{x'}$ and f_x are Shannon expansion co-factors,

Given $f = (x + f_{x'})(x' + f_x)$, Show that $f = x f_x + x f_{x'}$

(5)

Q10: (a) Given f(a, b, c, d) = ab + b'cd + acd
Show that f(a, b, c, d)= ab + b'cd using Shannon expansion and implement using 2:1 multiplexer and logic gates.

(b) f(a, b, c, d) = (a + b + c)(a' + d)(b + c + d)
Show that f(a, b, c, d)= (a + b + c)(a' + d) using Shannon expansion and implement using 2:1 multiplexer and logic gates.

(5+5)

Q11: Write two ways of describing a 4:1 multiplexer in Verilog HDL and also provide test bench for testing the same? Assume the following signal naming convention Input: D, control: S and output Y.

(3+3+4)