University of California at Berkeley
College of Engineering
Department of Electrical Engineering and Computer Sciences

EECS150 J. Wawrzynek
Spring 2009 5/14/09

Final Exam

Name: ANSWER KEY _____

ID number: _____

This is a *closed-book, closed-note* exam. No calculators or any other electronic devices, please.

The exam is made up of a series of true–false questions. Each question is marked with its number of points (one point per expected minute of time). A correct answer is worth that number of points. A blank answer is worth 0 points. *An incorrect answer is worth* **minus** *that number of points.* Therefore, if you don't know the answer, leave it blank, don't guess.

Read all the questions **before** you begin.

You can tear off the spare pages at the end of the booklet, use the backs of the pages, or the space provided by each question to work out your answers. Mark your answers with and "X" in the appropriate box.

Note: Unless noted otherwise, throughout this exam booklet we use the following notation: x' or $\overline{x}$ for logic inversion, $+$ for logic OR, $\oplus$ for exclusive-or, xy and $x \cdot y$ are equivalent and both mean x AND y.

Put your name and SID on each page.

Each of the following operator(s) form a complete Boolean algebra—i.e., they are able by themselves (along with constant 0 and 1) to represent any combinational logic function:

1. [1pt] AND with NOT — ☒ **T** ☐ **F**
2. [1pt] exclusive-NOR — ☐ **T** ☒ **F**
3. [1pt] 2-input MUX — ☒ **T** ☐ **F**
4. [1pt] 3-input NAND — ☒ **T** ☐ **F**
5. [1pt] 4-input XOR — ☐ **T** ☒ **F**
6. [1pt] Muller c-element — ☐ **T** ☒ **F**
7. [4pt] Given the following true-table, where "-" means don't care, the two logic expressions are equivalent to the true-table.

abc	f
000	0
001	-
010	-
011	1
100	-
101	0
110	1
111	0

$$f = b(a' + c'), \quad f = a'b + ac'$$

☒ **T** ☐ **F**

8. [1pt] The truth-table for an n-bit *carry-save adder* has exactly 2^{2n+1} rows. — ☐ **T** ☒ **F**

9. [4pt] Using only 2 and 3-input CMOS logic gates, with the cost per transistor equal to 1, the minimum cost for the carry function in a full-adder cell is 16.

☐ **T** ☒ **F**

10. [3pt] In Boolean algebra distribution of "+" over "$\oplus$" holds—i.e., $a + (b \oplus c) = (a + b) \oplus (a + c)$.

☐ **T** ☒ **F**

11. [3pt] In Boolean algebra distribution of "$\cdot$" over "$\oplus$" holds—i.e., $a(b \oplus c) = ab \oplus ac$.

☒ **T** ☐ **F**

12. [2pt] There exists 65536 unique Boolean functions of 4 variables.

☒ **T** ☐ **F**

13. [3pt] $(x' + y)(x + y') = x' \oplus y'$.

☐ **T** ☒ **F**

14. [4pt] For some function f, if $\overline{f} = ab + bc$, then $f = a'(b'c + bc') + a(b'c' + b'c)$.

☐ **T** ☒ **F**

15. [4pt] Assuming equivalent costs for ANDs and ORs, the number of N variable Boolean functions where the canonical sum-of-products (SOP) form has the same implementation cost as the canonical product-of-sums (POS) form is 2N.

☐ **T** ☒ **F**

16. [5pt] $f = a'e'c' + de'b'c' + d'e'bc' + ec$ is in *minimal* SOP form.

☒ **T** ☐ **F**

17. [3pt] It is possible to compute the 2-input XOR function with 5 2-input NOR gates and no other gates.

☒ **T** ☐ **F**

18. [1pt] Moore machines always have more states than the equivalent Mealy machine.

☐ **T** ☒ **F**

19. [2pt] One-hot encoded state machines are never Mealy machines.

☐ **T** ☒ **F**

20. [4pt] The number of unique 2-state Moore machines that have a 1-bit wide input signal and a 1-bit wide output signal is 16.

☐ **T** ☒ **F**

21. [5pt] In a 12-bit carry-select adder, grouping the bits into 3 groups of 4 bits each (4-4-4) results in lower overall delay than with any other arrangement of group sizes. Assume that the delay through a 2-input multiplexor is the same as the delay through a full-adder cell.

☐ **T** ☒ **F**

22. [5pt] Consider the design of a "2-bit-wide" serial adder used for adding 2 N-bit numbers (N is guaranteed to be even and at least 2). The 2-bit-wide adder is similar to the bit-serial adder from class, except that it adds *two* bit positions from each input within one cycle, instead of one; therefore it might need a longer clock period. Your design can only use flip-flops ($\tau_{clk-to-q} = \tau_{setup} = 1ns$), and full-adder cells ($\tau_{full-adder-cell} = 2ns$).

    On N-bit addition, using the 2-bit design, it is possible to achieve an add *latency* (in ns) less than that for the normal serial adder.

    ☒ **T** ☐ **F**

23. [5pt] Consider the design of a Moore style FSM that looks for an input pattern with alternating 0's and 1's starting with 0 (010101...). A reset signal is used to force the machine into a starting state and the output to 0. On the next cycle the FSM starts looking for the desired pattern (starting with a single 0), and outputs a 1 as long as the input is the proper pattern. If the input deviates from the desired pattern, the output becomes 0, until the desired pattern is seen again (starting with a single 0).

    It is possible to design an FSM with the desired behavior with fewer than 4 states.

    ☒ **T** ☐ **F**

24. [2pt] As a function of the number of input bits, N, the cost in terms of total transistors of a carry-lookahead adder is $\propto N + log_2(N)$.

    ☐ **T** ☒ **F**

25. [4pt] You are requested to perform a multiplication of 2 N-bit numbers, but your multiplication circuit only accepts N/2 bit wide inputs. You also have an adder circuit (of whatever width needed). The total number of times you will need to use the multiplier for the N * N multiplication is 3.

☐ **T** ☒ **F**

26. [3pt] -7/8 in 4-bit binary 2's complement representation is 1.001

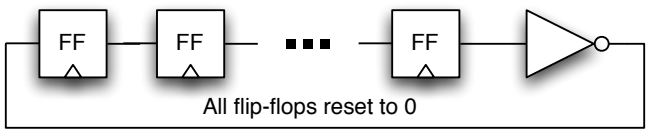
☒ **T** ☐ **F**

27. [2pt] A combinatorial (single-cycle) multiplication of an N-bit variable by a constant in the worst-case requires no more than $N/2 - 1$ adders/subtractors.

Omitted

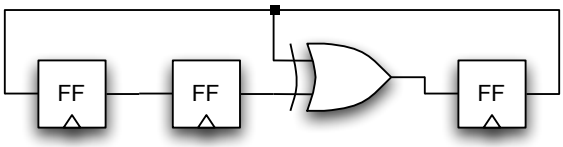
28. [3pt] The circuit shown below is an "inverted feedback ring counter". Such a counter with $2 \cdot log_2(M)$ flip-flops can generate M unique states.

☐ **T** ☒ **F**



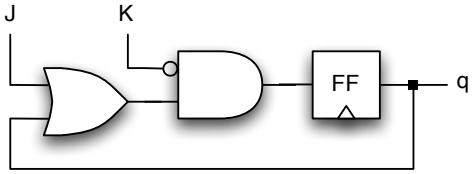
29. [4pt] The LFSR circuit shown below can be used to generate a pseudo-random sequence with the maximum period achievable for a 3-bit LFSR.

☒ **T** ☐ **F**



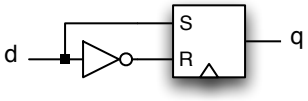
30. [2pt] The circuit shown below correctly functions as a JK-flip-flop.

☐ **T** ☒ **F**



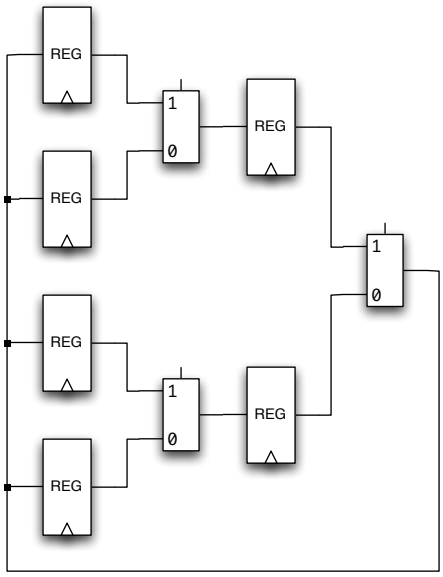
31. [1pt] The circuit shown below correctly functions as a D-flip-flop.

☒ **T** ☐ **F**



32. [4pt] Given the datapath below, the minimum number of cycles needed to exchange the top-left-most register value with the bottom-left-most register value is 4.

☐ **T** ☒ **F**



33. [4pt] You need to compute the polynomial, Ax^3+Bx^2+x, for 4 values of x, using 2 multipliers and one adder. The multipliers and the adder are not pipelined, i.e., each operate in one cycle. This will take a minimum total of 10 cycles.

☒ **T** ☐ **F**

For the next two questions ... An FPGA chip is made up of 4-LUTs, flip-flops, and switches. Switches are used to connect LUTs to other LUTs, LUTs to other flip-flops, flip-flops to other LUTs, or flip-flops to other flip-flops. For the flip-flops, $\tau_{clk-q} = \tau_{setup} = 100ps$. The LUT delay is $\tau_{LUT} = 200ps$, and the switch delay is $\tau_{switch} = 100ps$. Ignore the wiring delay.

34. [2pt] The minimum possible period of a ring oscillator built in this FPGA is 600ps.

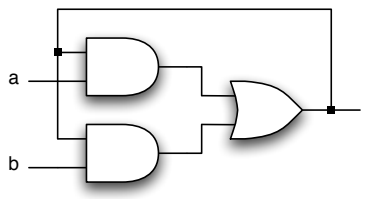
☐ **T** ☒ **F**

35. [5pt] A 4-bit binary counter implemented on this FPGA will have a minimum clock period of 600ps.

☒ **T** ☐ **F**

36. [2pt] The following is a correct implementation of a Muller c-element.

☐ **T** ☒ **F**



37. [4pt] You are required to design a special 4-input multiplexor using as few CMOS transistors as possible. As is usually the case with a 4-input multiplexor, it has 4 data inputs and two control inputs. However, in this case one of the data inputs is permanently connected to logic 1 and another of the data inputs is permanently connected to logic 0.

    It is possible to design this circuit with a total of 16 transistors.

    ☒ **T** ☐ **F**

38. [1pt] The following Verilog code synthesizes to a right shifter followed by a register:

```
module foo (A, B, C);
input A, B;
output C;
reg C;
always @ (*)
  C = A >> B;
end module
```

    ☐ **T** ☒ **F**

39. [4pt] Imagine that you are working for a company and trying to decide between using an FPGA or an ASIC in your product. The FPGA you are considering has a per chip cost of $100, and by using it you will incur a non-recurring engineering (NRE) cost of $1M. The ASIC per chip cost is $10 and has an NRE cost of $4M. Your sales department tells you they expect to sell 50K units of your product. The product requires either one FPGA or one ASIC. Your goal is to save costs for your company.

    Your best choice is the FPGA.

    ☐ **T** ☒ **F**

40. [4pt] Consider a *single-cycle* MIPS processor with only four instructions, add, sub, lw, and sw, running programs where each of the four instructions occur with equal probability. If it were practical to do so, varying the clock period on a per instruction basis would result in a overall performance improvement of at least 10%.

☒ **T** ☐ **F**

41. [4pt] *Multi-threading* is a known way to help eliminate hazards in pipelined processors. Instructions from multiple independent programs (or threads) are interleaved in the pipeline. This is equivalent to the "C-slow" technique we presented for optimal pipeline utilization.

    Consider a standard 5-stage MIPS pipeline implementation, but with no bypassing or forwarding circuitry. Multithreading this processor with 4 independent interleaved instruction streams is sufficient to eliminate all control and data hazards.

☒ **T** ☐ **F**

42. [1pt] I wrote my SID on all pages.

☒ **T** ☐ **F**

Spare page. *Will not be graded.*

Spare page. *Will not be graded.*

Spare page. *Will not be graded.*