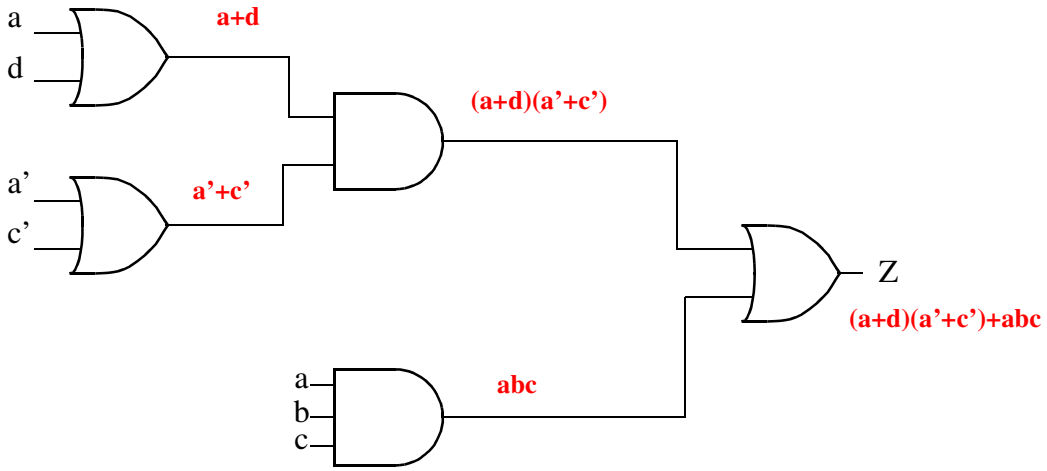
CPE/EE 422/522 **EXAM 2** **Name**_____**KEY**_____
Class _____

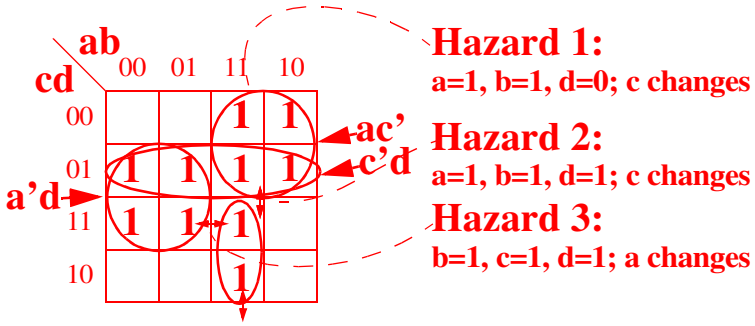
Spring Semester 2009

Work should be performed systematically and neatly with the final answer being underlined. This exam is closed book, closed notes, closed neighbor. Allowable items on desk include: exam, data manual, pencils, straight edge, and calculator. All other items must be removed from student's desk. Students have approximately 90 minutes (1 1/2 hours) to complete this exam. Best wishes!

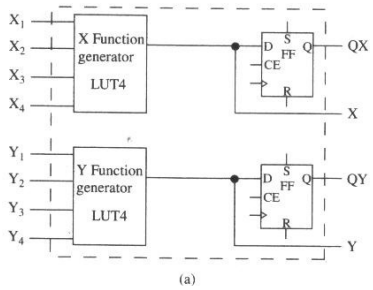
1. [10 points] For the network shown below, find all static 1-hazards. For each 1-hazard found, specify the conditions which will cause the hazard to appear at the output (i.e. specify the logic values of the variables which are constant and clearly specify the variable that is assumed to be changing). If there are no 1-hazards found, use a K'map to show why this is the case.



$Z^T = (a+d)(a'+c')+abc = aa' + ac' + a'd + c'd + abc$



2. [22 points] An 8-bit right shift register with parallel load is to be implemented using an FPGA with logic blocks with the configuration shown below:

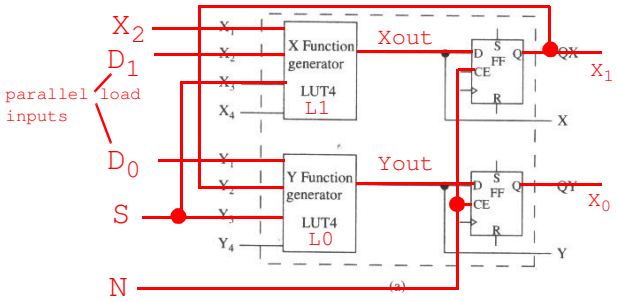


The flip-flops are labeld $X_7X_6X_5X_4X_3X_2X_1X_0$. The Control Signals N and S operate as follows: N=0, do nothing; NS = 11, right shift; NS=10, load. The serial input for right shift is SI.

a) How many logic blocks are required (minimum number? (Assume N is connected to CE through routing paths that are not shown).

4 Logic Blocks minimum since input N is connected to CE of the flip-flops. Note: each LUT4 implements 3 variable function that allows data to be loaded from associated parallel bit or obtained from lower-ordered flip-flop in the shift register chain.

b) Show the required connections for the rightmost (least significant block) that is shown below. Assume N is connected to CE as discussed above.



c) Give the Boolean equations for the function generator (LUT) outputs for this rightmost block.

X Function LUT4 (i.e. L1):

$Xout = S*X_2 + \overline{S}*D_1$

Y Function LUT4 (i.e. L0):

$Yout = S*X_1 + \overline{S}*D_0$

3. [5 points] What is the primary difference between sequential logic and combinational logic?

In combinational logic the current value of the outputs depend only upon the current values on the inputs. In sequential logic the current value of the ouputs not only depend upon the current value of the inputs but on past values of the inputs. Sequential logic has memory whereas combinational logic is memoryless.

4. [15 points] Refer to the handout on Laboratory 4 and answer the following questions.

a) What was the overall purpose of the CLOCK_DIVIDE module? Why was this module necessary?

The DS1620 device has a maximum clock rate that is much lower that the 50 Mhz, or 27 Mhz, clocks provided by the DE-2 board. The CLOCK_DIVIDE module is designed to slow down the clock that goes to the DS1620 to an acceptable level.

b) What was the function of each of the CLK_OUT, RST, DQ signals that drove and/or received data from the DS 1620 IC?

CLK_OUT?

Provide a controlled clock to the DS 1620. The clock signal should pass through the input clock (CLK_IN) for a portion of the cycle, and be placed at a logic high during another portion of the cycle. During this portion of time (when the CLK_OUT is high) it also pulses low one time to initiate the DS 1620's temperature conversion circuitry.

RST?

This is the major control signal that goes between the digital design and the external DS 1620 device. This signal is high during the peripheral control portion of the overall cycle and is a logic low during the slave mode portion of the cycle.

DQ?

This is the bidirectional signal that allows data to be sent and received from the DS 1620 in a serial manner. Eight bit command data is sent to the DS 1620 and in this case nine bit temperature data is received from the DS 1620 using this signal.

c) What was the overall purpose and function of the DS1620_INTERFACE module?

DS1620_INTERFACE module is designed to produce the timing diagram shown in Figure 2 of the Handout. It activates the CLK_OUT in a manner that passes through the CLK_IN signal during portions of the cycle and brings the CLK_OUT signal high during the other portion of the cycle (except for a single temperature conversion cycle as discussed above. It also drives the DQ_OUT signal with the Read Temperature Command and receives temperature data from the DQ_IN signal during the associated periods of the cycle.

d) Briefly describe how you implemented this module in your design [Specify how you sent and received data (on what clock edge) and the number processes you used].
Many answers possible. I created a two process implementation with one process reading the data on the rising edge of the system clock and the other process driving the RST, DQ_OUT, signals on the falling edge of the same clock. Triggering on both edges of the clock allowed for the read to be performed in the middle of the DS 1620 output period. A concurrent assignment statement was used to enable the clock echo from CLK_IN to CLK_OUT. To control this Gating an internal signal was ORed with the CLK_IN line. This signal was controlled by the process that triggerred on the rizing edge of CLK_IN. It was set to '1' during periods of the clock that were not to echo the clock pulse and set to '0' at periods where the clock pulse was to be passed through.

e) What was the purpose of the TEMP2BCD module?
To output the temperature in Binary Coded Decimal format. It is capable of displaying temperature in either Fahrenheit or Celsius format to the 1/2 degree. This is actually a large lookup table that produces two 4-bit BCD values, a single bit value when the number exceeds 100 degrees and another single bit value for the 1/2 degree.

f) What was the purpose of the BINTOHEX module?
Convert from 4 bit binary (or BCD) value over to a 7 bit form that will display the correct character on the seven segment LEDs.

e What was the purpose of the FAR_CEL, and CLR inputs?
These inputs allowed the TEMP2BCD module to display in either Fahrenheit or Celsius.

5. [8 points] What device does the following VHDL model represent? Be as specific as possible.

```
process(CLK, RST)
  variable Qtmp: bit;
begin
   if RST = '1' then Qtmp := '0';
   elsif CLK'event and CLK = '1' then
      if T = '1' then
         Qtmp := not Qtmp;
      end if;
   end if;
   Q <= Qtmp;
end process;
```

This is most likely a rising edge clocked T flip-flop with asynchrounous reset

6. [20 points] Use Shannon's expansion theorem around **a** and **b** for the function

Y = abcde + cde'f + a'b'c'def + bcdef' + ab'cd'ef' + a'bc'de'f + abcd'e'f

so that it can be implemented using only four-variable function generators (4 variable LUT). Draw a block diagram to indicate how **Y** can be implemented using only four-variable function generators. Indicate the function realized by each four-variable function generator.

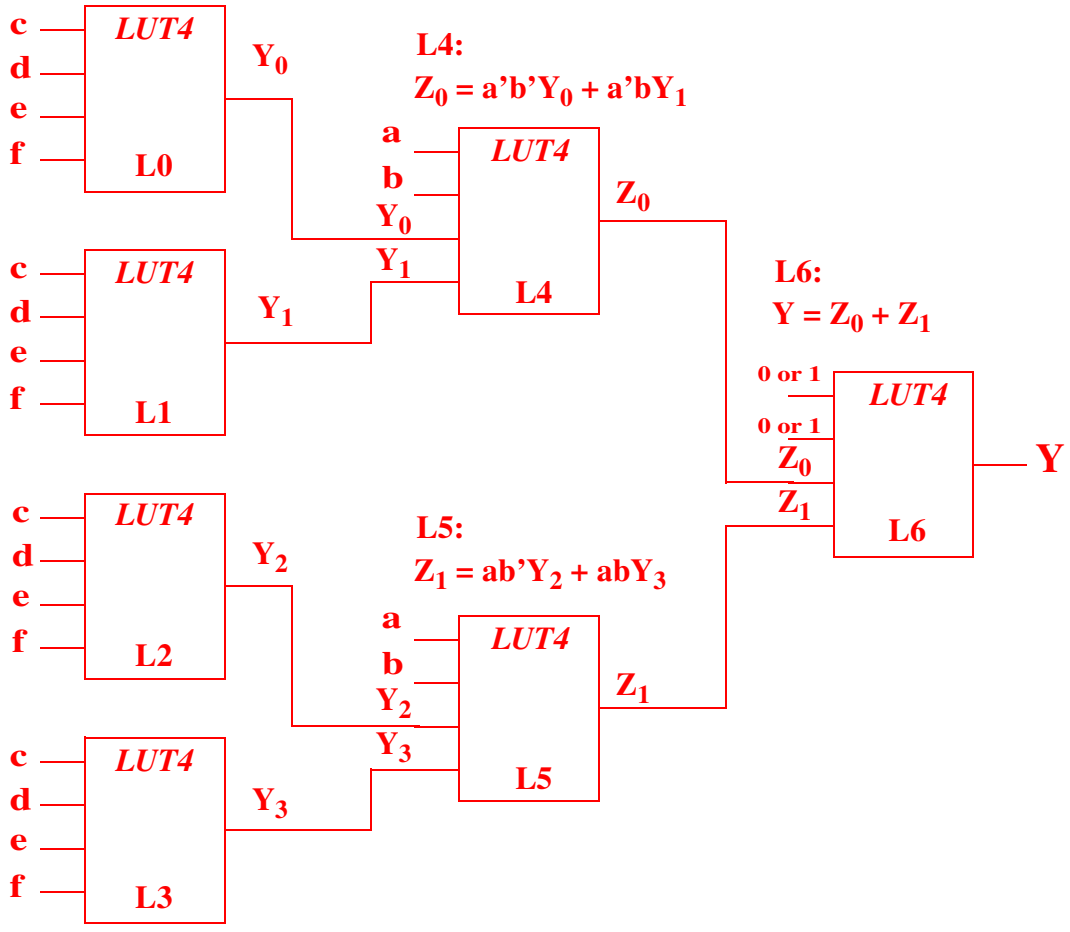
$Y = a'b'Y_0 + a'bY_1 + ab'Y_2 + abY_3$

L0: *(a=0, b=0);*
$Y_0 = cde'f + c'def$

L1: *(a=0, b=1);*
$Y_1 = cde'f + cdef' + c'de'f$

L2: *(a=1, b=0);*
$Y_2 = cde'f + cd'ef'$

L3: *(a=1, b=1);*
$Y_3 = cde + cde'f + cdef' + cd'e'f$



7. [10 points] When the following VHDL code is simulated, A is changed to '1' at time 5 ns. Complete the table below that shows all changes in A, B, and D and the times at which tye occur through time = 40 ns. Include all delta delays.

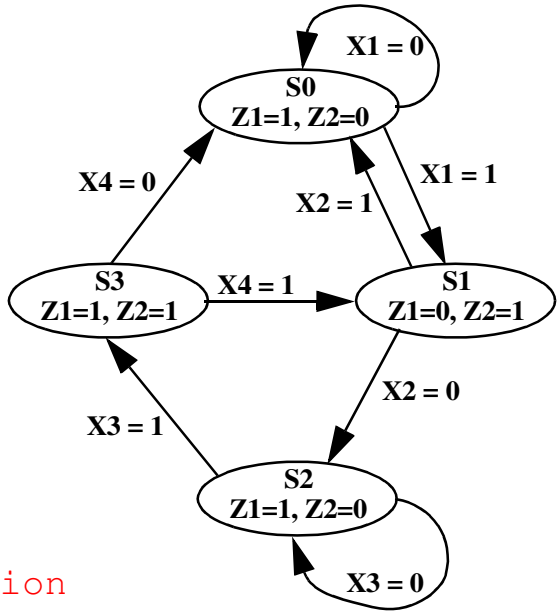
```
entity Q1F00 is
   port(A: buffer bit);
end Q1F00;

architecture Q1F00 of Q1F00 is
   signal B, D: bit;
   begin
      D <= A xor B after 10 ns;
      process(D)
         variable C: bit;
      begin
         C := not D;
         if C = '1' then
            A <= not A after 15 ns;
         end if;
         B <= D;
      end process;
end Q1F00;
```

Time	A	B	D
0 ns	0	0	0
5 ns	1	0	0
15ns	1	0	1
15 ns + Δ	1	1	1
25 ns	1	1	0
25 ns + Δ	1	0	0
35 ns	1	0	1
35 ns + Δ	1	1	1
40 ns	0	1	1

8. [10 points] For the given state graph, derive the simplified next-state and output equations by inspection. Use the following one-hot state assignments for the flip-flops $Q_0Q_1Q_2Q_3$: S0, 1000; S1, 0100; S2, 0010; S3, 0001;

Inputs:
X1, X2, X3, X4

Outputs:
Z1, Z2



By inspection

Next State Equations:

$Q_0^+ = Q_0\overline{X1} + Q_1X2 + Q_3\overline{X4}$

$Q_1^+ = Q_0X1 + Q_3X4$

$Q_2^+ = Q_1\overline{X2} + Q_2\overline{X3}$

$Q_3^+ = Q_2X3$

Output Equations:

$Z1 = Q_0 + Q_2 + Q_3 = \overline{Q}_1$

$Z2 = Q_1 + Q_3$