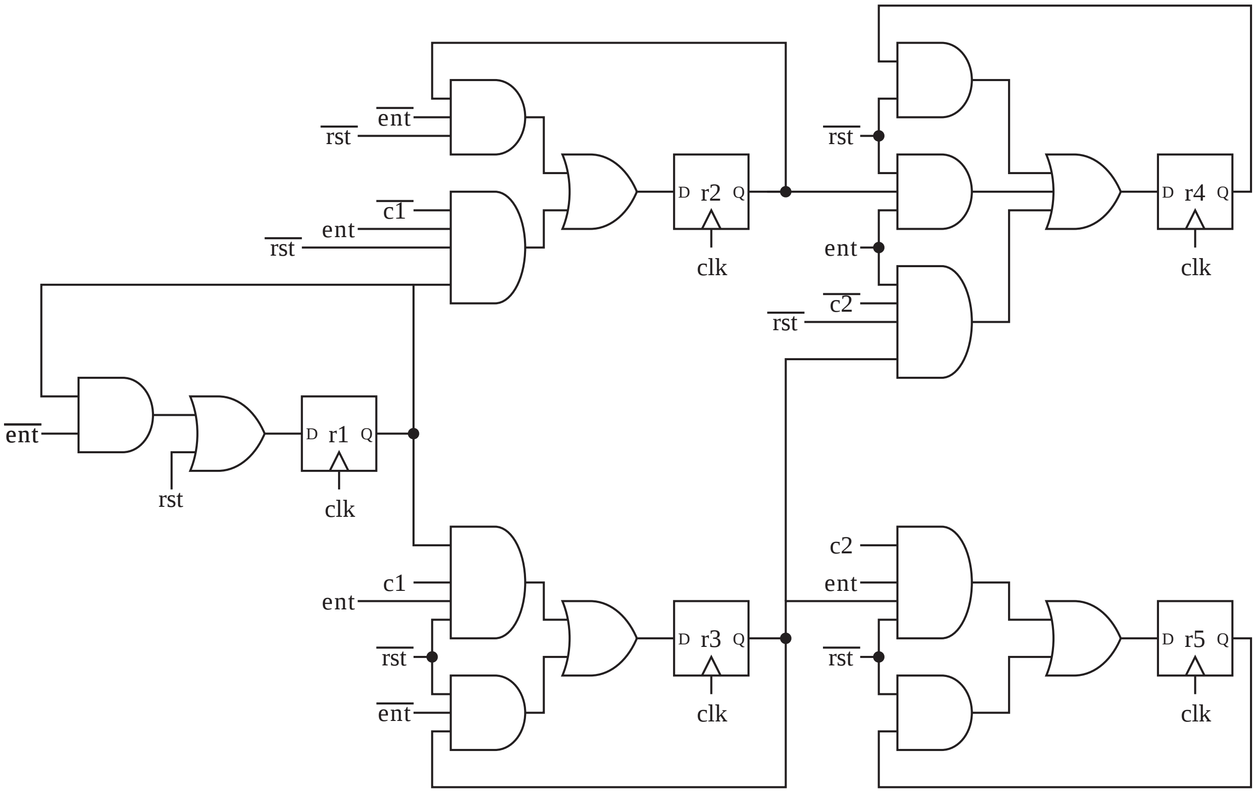
EECS 151/251A Homework 3

Due Monday, Feb 13th, 2023

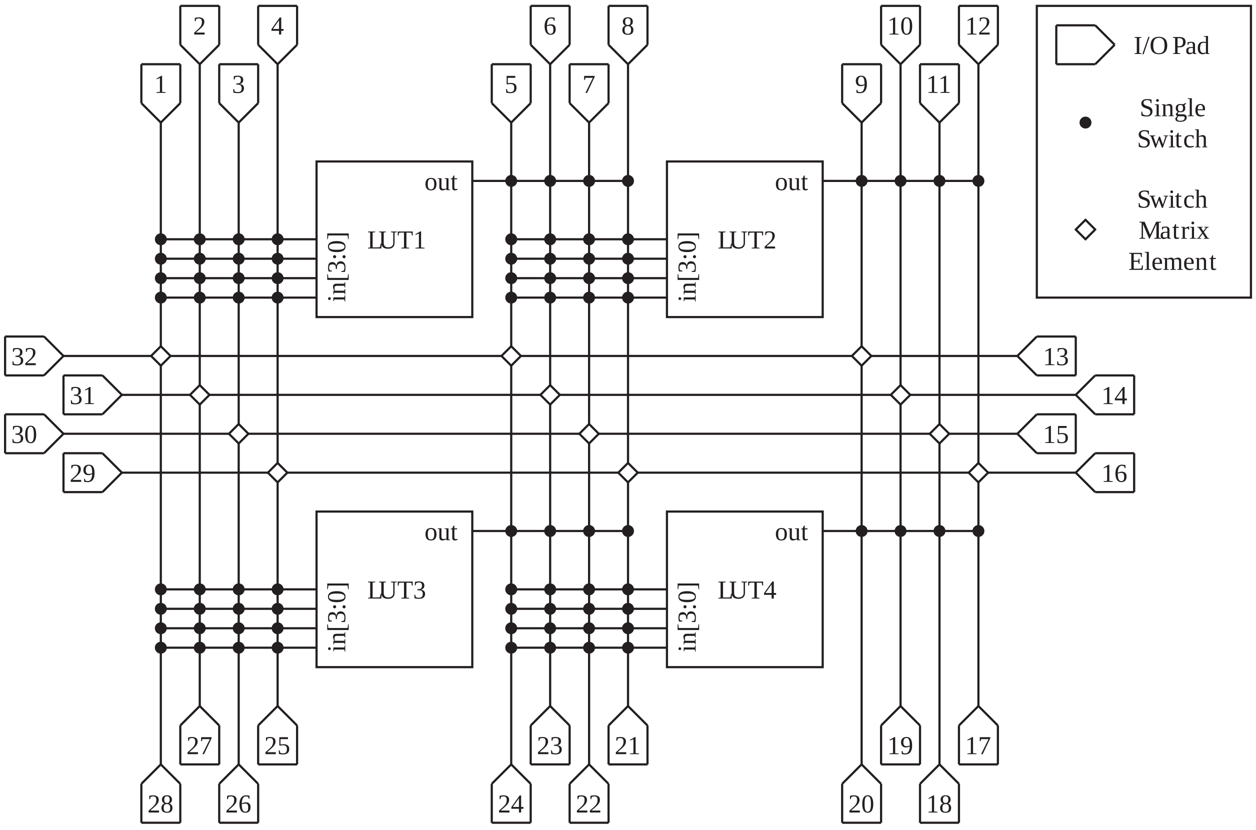
Problem 1: LUT Mapping

Imagine you have an FPGA consisting of logic blocks each of which contains one 5-LUT and one FF. Partition the following circuit of four inputs {*ent*, *rst*, *c*1, *c*2} and describe the function of each LUT in Boolean expression (you don't have to simplify). The FFs are named {*r*1, *r*2, *r*3, *r*4, *r*5}, and you can use these names to represent their outputs.



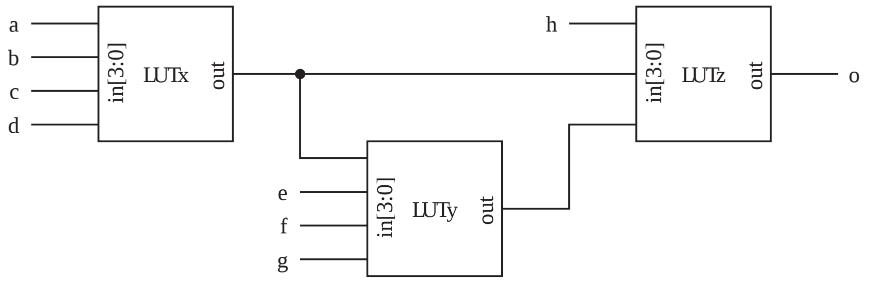
Problem 2: FPGA Placement and Routing

A 2 × 2 4-LUT FPGA is shown below. In this FPGA, each LUT takes input from one side and generates output to the other side. Switches are implemented as a single transistor that joints or disjoints two wires depending on its configuration bit. Each switch matrix element contains six switches, one for each pair of wire segments. External I/Os are assigned to the I/O pads according to the configuration. Unassigned I/O pads have infinite impedance (no current flows into it).



(a) What is the minimum number of switches needed to connect the output of LUT4 to the input of LUT1. (*Hint*: We need at least three switches to connect the output of LUT1 to the input of LUT4)

(b) *Critical path delay* is the maximum delay between input and output for a given placement and routing. Let delay of each switch be 1 and delay of each LUT be 4. Given a LUT mapping below, when we place LUTx to LUT1, LUTy to LUT2, and LUTz to LUT3, while assigning *a*-*d* to pad 1-4, *e*-*g* to pad 5-8, *h* to pad 25, and *o* to 21, what will be the critical path delay? (*Hint*: delay from *h* to *o* is 6.)



(c) Is there a better placement and routing in terms of critical path delay? If any, write it down and report its critical path delay. Otherwise explain why.

(d) We cannot place LUTx to LUT2 and LUTy to LUT1. Why?

Problem 3: Bit-Stream Reverse Engineering

Imagine you obtained a bit-stream from somewhere and want to figure out what it is.

1. The bit-stream is `0x1777A5A5965A9696`. This bitstream is fed in from right to left (i.e. `6` is fed first and `1` is fed last).
2. Every LUT in the FPGA has N inputs (the value of N is part of the mystery). The LUTs are numbered 0, 1, 2, ...
3. Each LUT has an output labeled y_i and inputs labeled x_{i_j}, where i is the LUT number and j is the input number.
4. For programming, the LUTs are connected in a shift register. The bit-stream will be shifted in from LUT0 and then pushed to the subsequent LUTs. (This implies that the left most `1` must be part of the function for LUT0.)
5. The shift register is ordered in the ascending order of input for each LUT. That is, the first register stores the output for an input 00...0, the second register stores the output for an input 00...01 (only x_{i_0} is 1), and so on. (This implies that if $N = 2$, LUT0 programmed with `1` works as an AND gate.)
6. One of the LUTs is programmed as a 2-input gate. Furthermore, it is the only LUT programmed as a 2-input gate.

(a) How many LUTs would the above bit stream program?

(b) Write down the function of each LUT in Boolean expression (no need to simplify but notice there are some patterns that can be concisely expressed with XORs).

251A only — *Optional* Challenge Question for 151

After a while, you realized it takes two M-bit inputs a and b (M is unknown) and calculates sum of a, b, and some constant c (the result may be truncated). What is c?

Problem 4: Functional Completeness Property

Prove that a collection of 2-to-1 multiplexers can implement any Boolean function. Assume constant 0 and 1 are freely available.

Problem 5: Boolean Algebra

Prove the equivalence of Boolean expressions through transformation for each of the following. Show all steps where you used postulates or theorems other than associative and commutative laws. Also write down the postulate or theorem you used for each of those steps. (Answers using truth tables are not acceptable.)

(a) $x + \overline{x}y = x + y$

(b) $xy + \overline{x}z + yz = xy + \overline{x}z$

(c) $\overline{\overline{x} \cdot \overline{y} + z} + z + xy + wz = x + y + z$

Problem 6: K-Maps

Consider a 4-input function represented by the following truth table.

a	b	c	d	x
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(a) Derive the minimul SOP form using a K-map. Also, how many 2-input gates do you need to directly implement that SOP? (An N-input AND/OR gate requires $(N-1)$ 2-input AND/OR gates.)

(b) Implement this function using only three 2-input gates.

Problem 7: K-Maps with Don't-cares

The function of a 2-bit encoder is shown below. - stands for Don't-care. Derive the minimal SOP (in terms of number of products) for each output using a K-map.

a	b	c	d	x	y
0	0	0	1	0	1
0	0	1	0	0	0
0	1	0	0	1	0
1	0	0	0	1	1
others				-	-

Problem 8: NOR/NAND Network

A NOR network is a gate-level circuit that uses only NOR gates. NAND networks use NAND gates instead. Draw a functionally equivalent NOR network and NAND network for the following circuit. (*Hint*: You may use 1-input NOR/NAND gates as inverters.)