# CDA 4213/CIS 6930 CMOS VLSI
# Fall 2019

## Final Project

## Due date(s)
## Partial Design Report:
Week of 18th November
## Final Design Report:
Monday, 9th December

| | |
|---|---|
| Today's Date: | 12/10/2019 |
| Your Team Name: | Joseph? |
| Team Members: | Joseph Daniel Moreno, U#60156204 |
| Work Distribution | Joseph: Drawing/planning, layout creation, simulation/testing |
| No. of Hours Spent: | 20 |
| Exercise Difficulty: (Easy, Average, Hard) | Hard |
| Any Feedback: | |

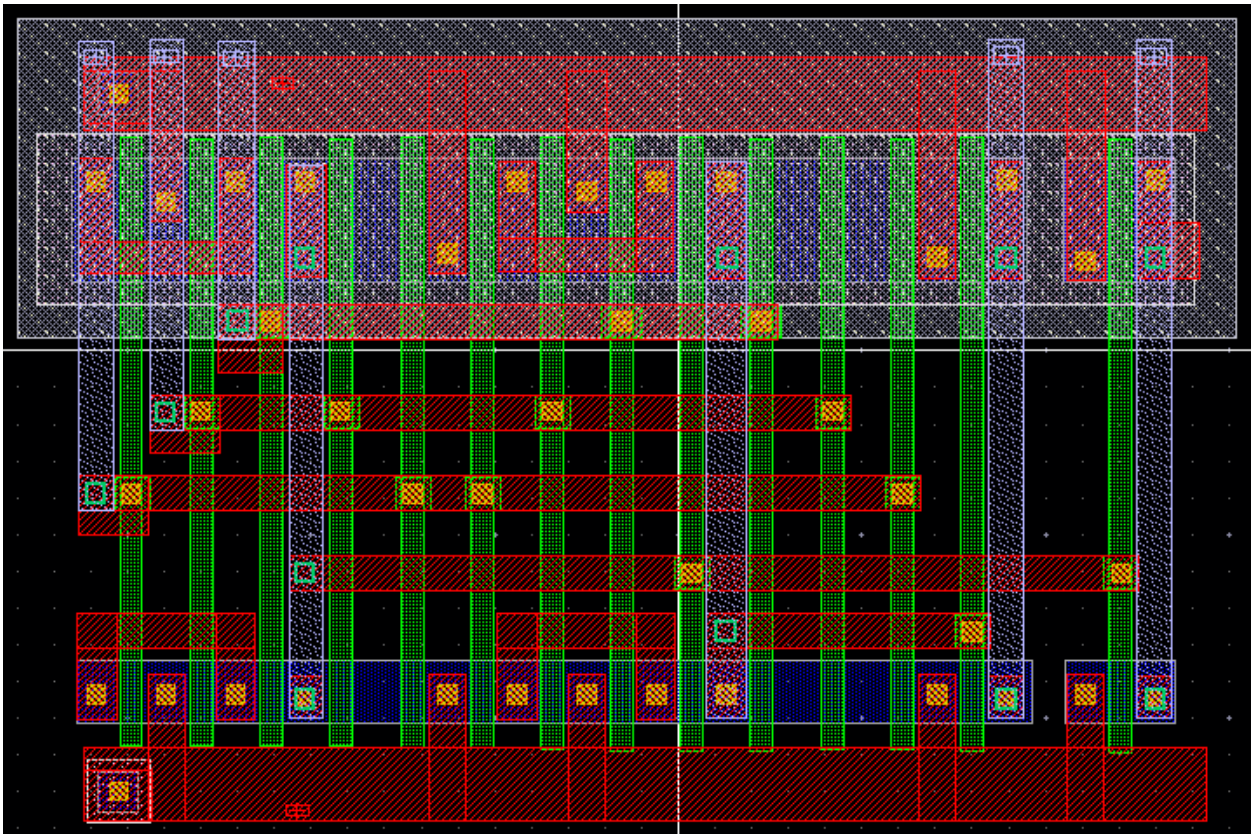
(1) **(10 pts)** Proposed Design – Bit slice design

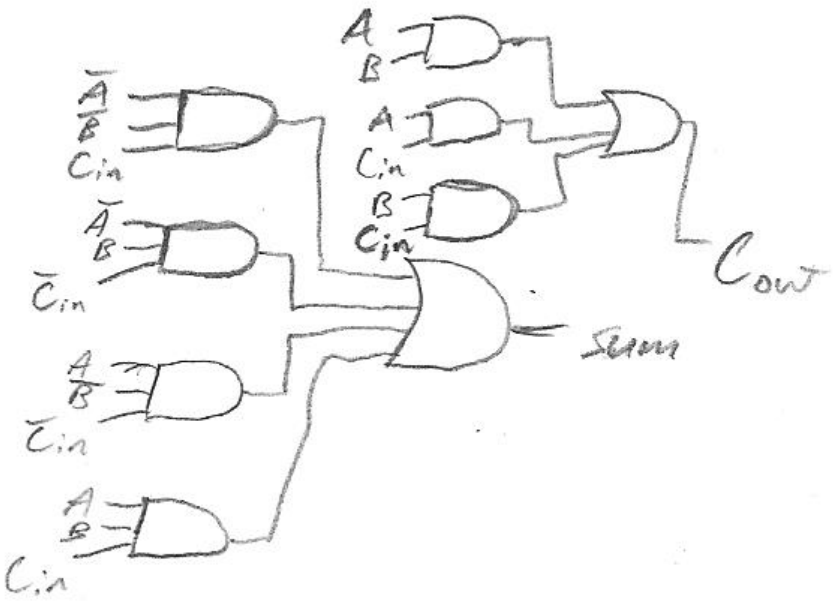
(a) List all module bit-slices you have used for your design.

Full adder, carry-save adder, 8-bit SIPO register, and 16-bit PISO register.

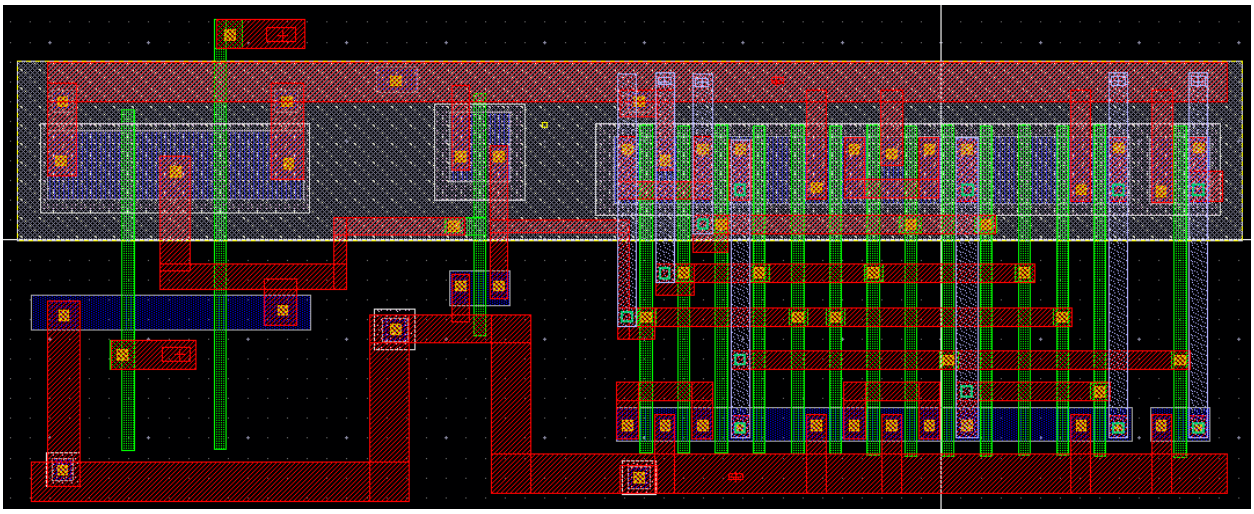
(b) For each bit slice, show the gate-level design and layout design. For layout, include the snapshot from Cadence Virtuoso. If you have used any other blocks, include them as well.

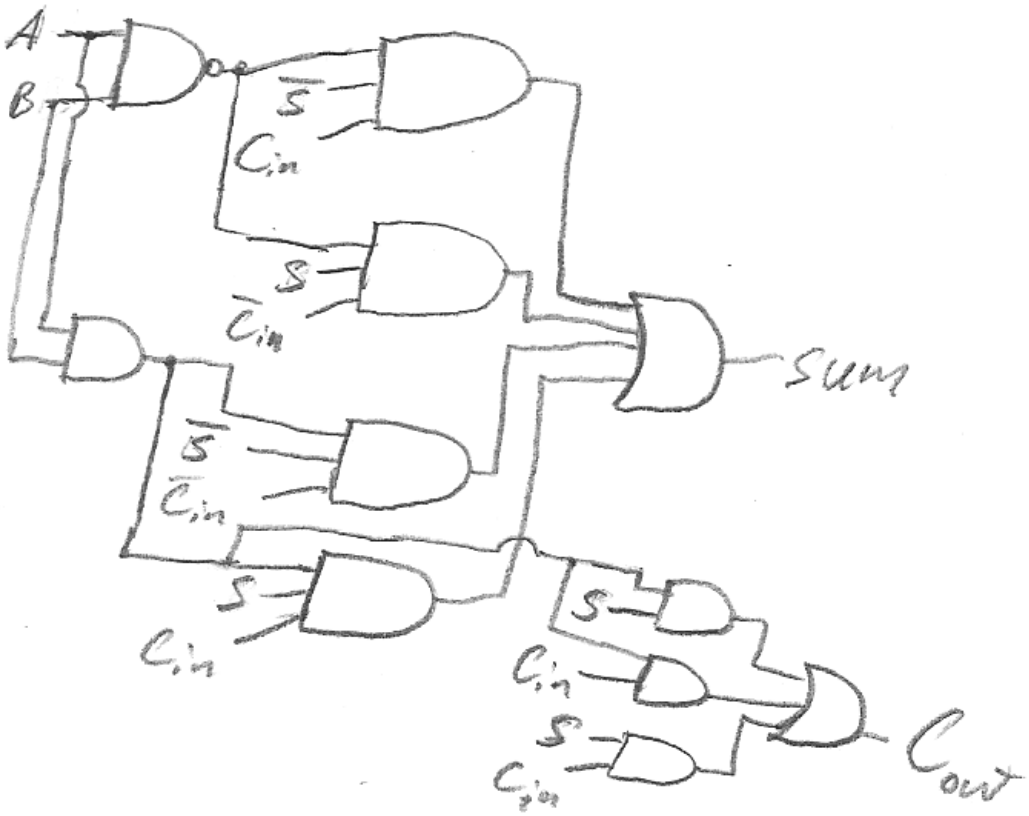
I. Full Adder:



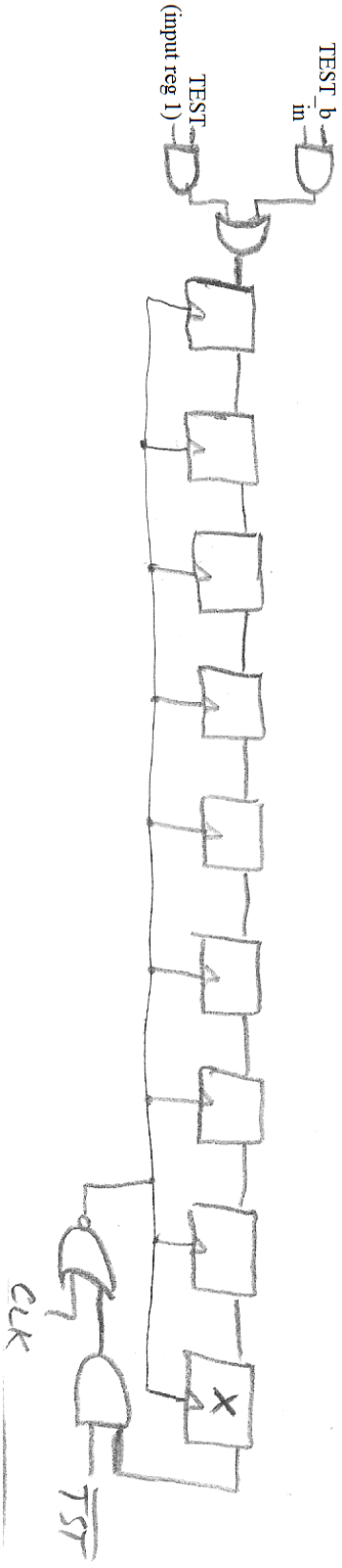


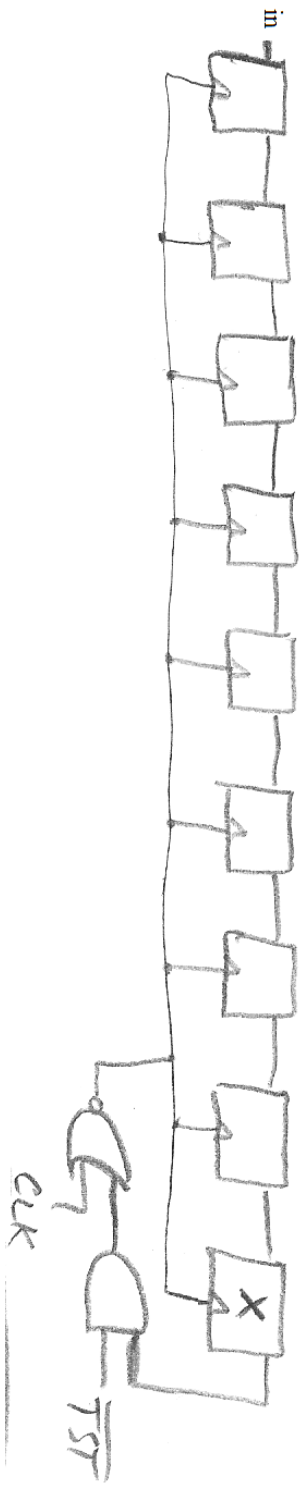
II. Full Adder with AND gate:

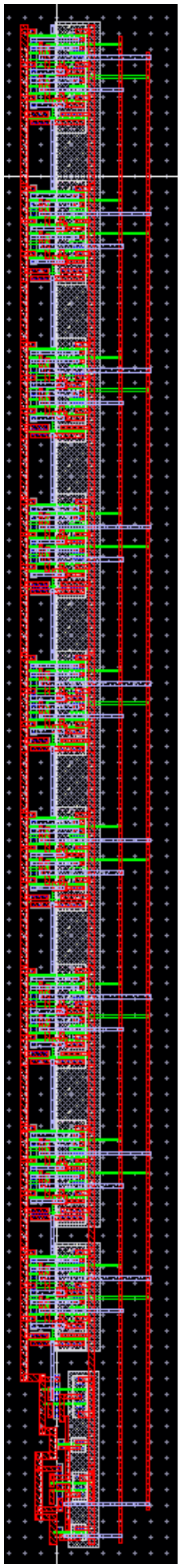




III. Registers (Inputs and Output):



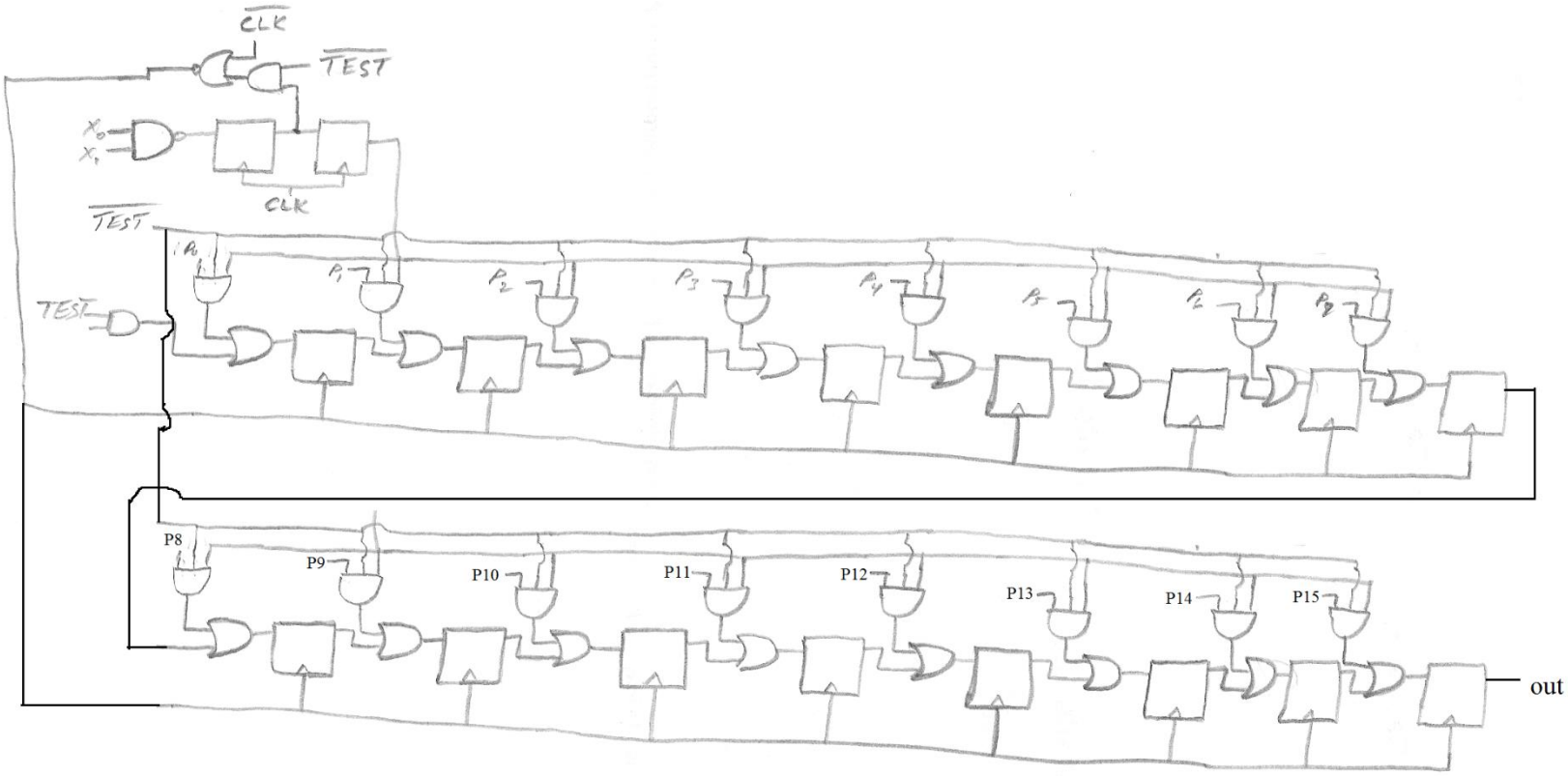



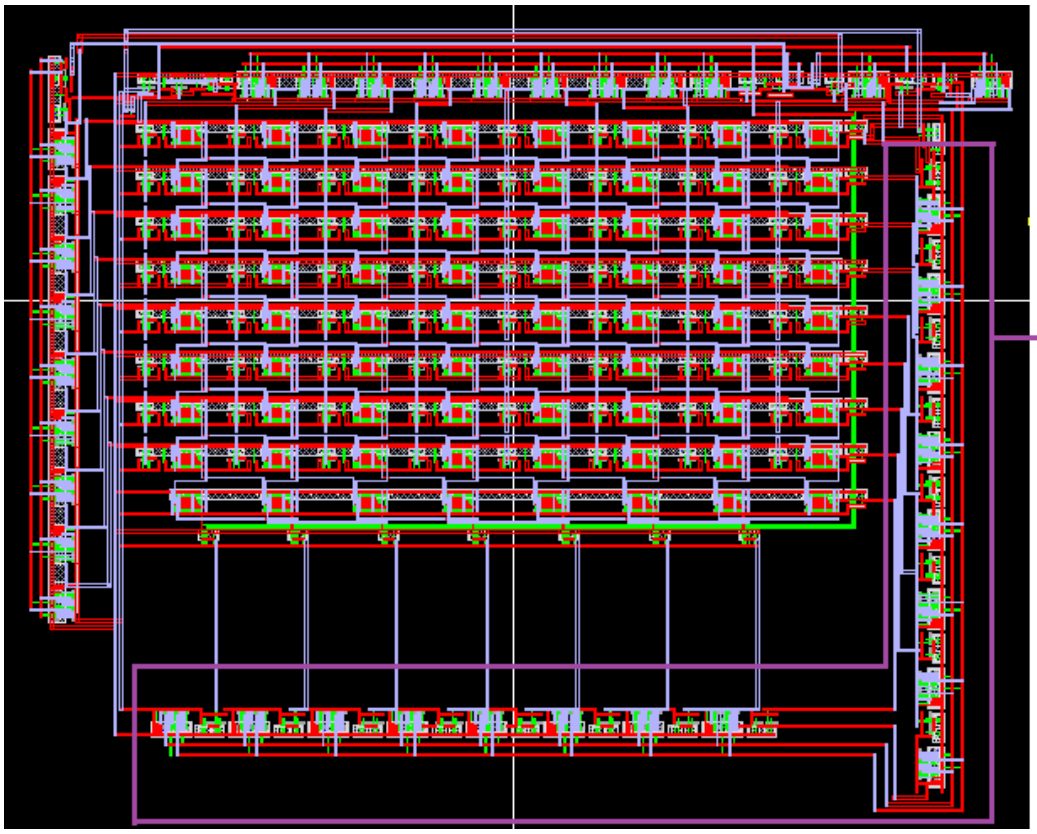



8-Bit SIPO Register

Rotated 90 deg. clockwise

16-Bit PISO Register



16-bit
PISO

IV. Ring Oscillator (If used):

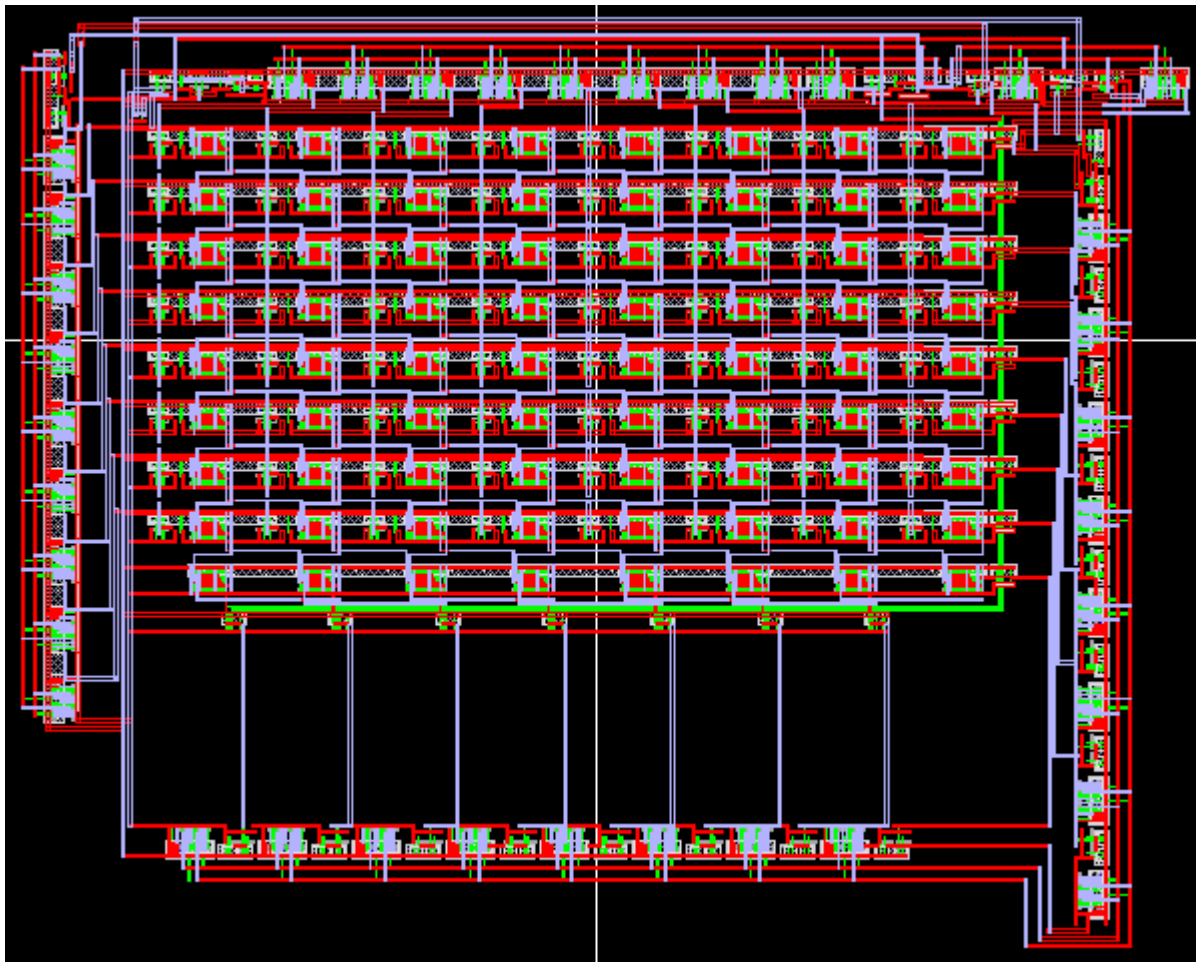
(2) **(15 pts.)** Show the layout of your multiplier with the registers (outside the padframe). Explain the design and functionality of your multiplier.

The 8 by 8 multiplier is made up of 8 rows and columns of carry-save adders with one more row of 8 full adders below. The product bits are output through the rightmost carry-save adders' and through the full adders at the bottom; least significant product bit is at the top-right and the most significant bit is at the bottom left.

The input registers to the left and at the top of the 8 by 8 multiplier are 8-bit SIPO. There is one additional register for each SIPO, reg_$X_0$ and reg_$X_1$, which are used to stop the clock from pulsing into the registers. When the reg_Xs = 1, the clock is disallowed from pulsing into the registers. Because of this, an extra logical 1 is required at the beginning of each input; if I want to use $(127)_{10}$ as input for the multiplier, then $(0111\ 1111\ 1)_2$ needs to be clocked into the register. TEST_b (or $\overline{TST}$ or $\overline{TEST}$; the inverse of the TEST input) is AND'd with each reg_X's output. Each AND output is fed into a NOR with the clock and those NOR outputs are fed into the SIPO's respective clock input. If TEST = 0 and reg_X = 1, the registers stop clocking in input. Once TEST = 1, the registers will continue clocking in input regardless of the reg_X's output; the SIPO at the top is cascaded with the SIPO on the left when TEST = 1.

The output register lining the right and bottom of the 8 by 8 multiplier is a 16-bit PISO. There are two additional registers, one for allowing the clock to pulse into the PISO, reg_CLK_EN, and the other is used to disallow input from the multiplier to clock into the PISO, reg_LOAD. reg_CLK_EN, takes in the NAND output of the reg_Xs. Similar to the purpose of the reg_Xs, when reg_CLK_EN = 0 it allows $\overline{CLK}$ to pulse into the output register; this is used to disallow any values from the multiplier to clock into the register while the multiplier is calculating the product. After one clock cycle of reg_CLK_EN becoming 0, reg_LOAD becomes 0; $\overline{CLK}$ is used for the output register so that the register loads in (parallel-in) values from the multiplier ($P_0, P_1, \dots, P_{15}$) on the negative edge of the clock cycle between reg_CLK_EN becoming 0 and reg_LOAD becoming 0. Once reg_LOAD = 0, the parallel input is cut off by a 3-input AND while the clock continues to shift the bits through the output register. If TEST = 1, $\overline{CLK}$ continues to pulse into the output register regardless of reg_CLK_EN, the parallel input is cut off by the 3-input AND regardless of reg_LOAD, and the output register is cascaded with the input registers to form a 32-bit shift register.

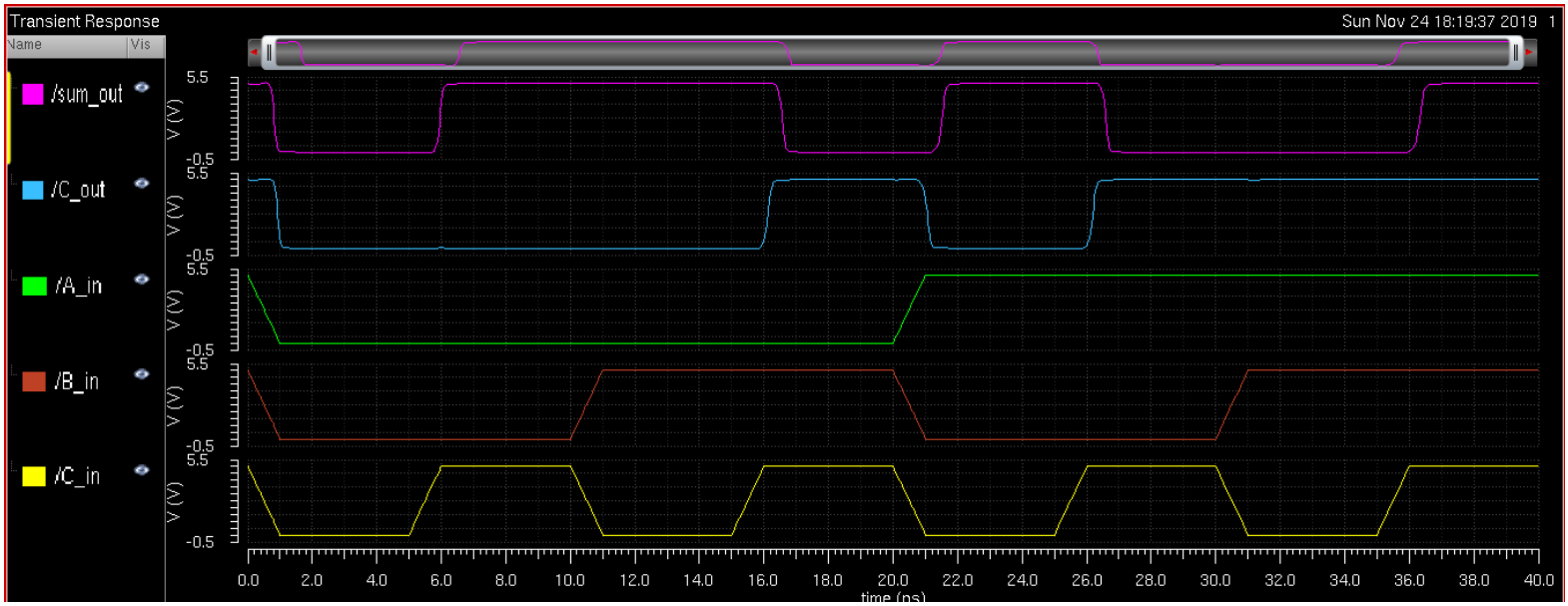
The final design's test mode does not work correctly because the input registers shift their bits on the positive edge of the clock while the output register shifts its bits on the negative edge. If TEST is turned from 0 to 1 right before a positive edge of the clock, the input registers will shift 1 bit over while the output register's bits sit still so the output register misses the shifted bit from the input registers. I observed this in simulation because a string of what should be 15 logical 1s from the input registers would come out as 14 logical 1s. Overall, turning TEST from 0 to 1 allows the clock pulse into the registers so that their contents may be flushed out, but the output is incorrect due to the design error.



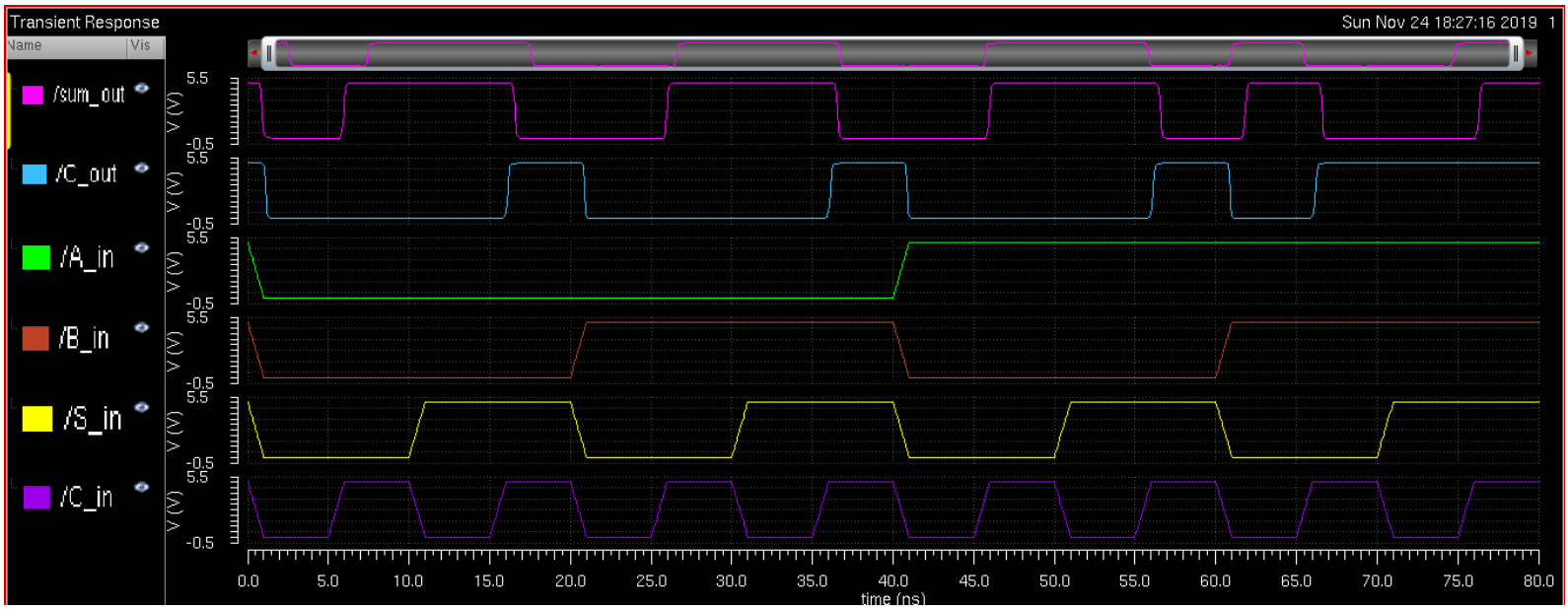
(3) **(25pts)** Simulation Results (without padframe):

(a) (5 pts total) Individual cells:

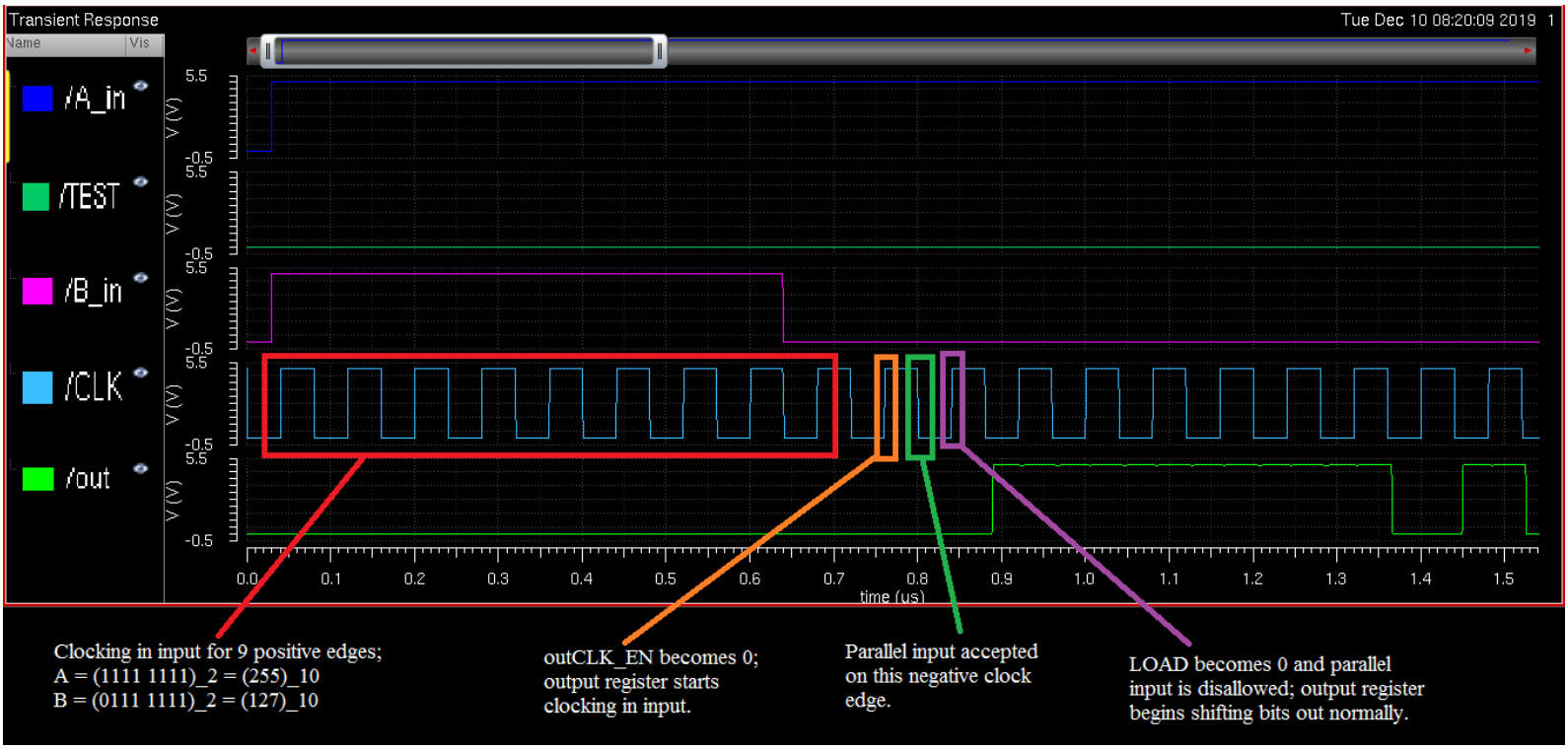
I. Full Adder:

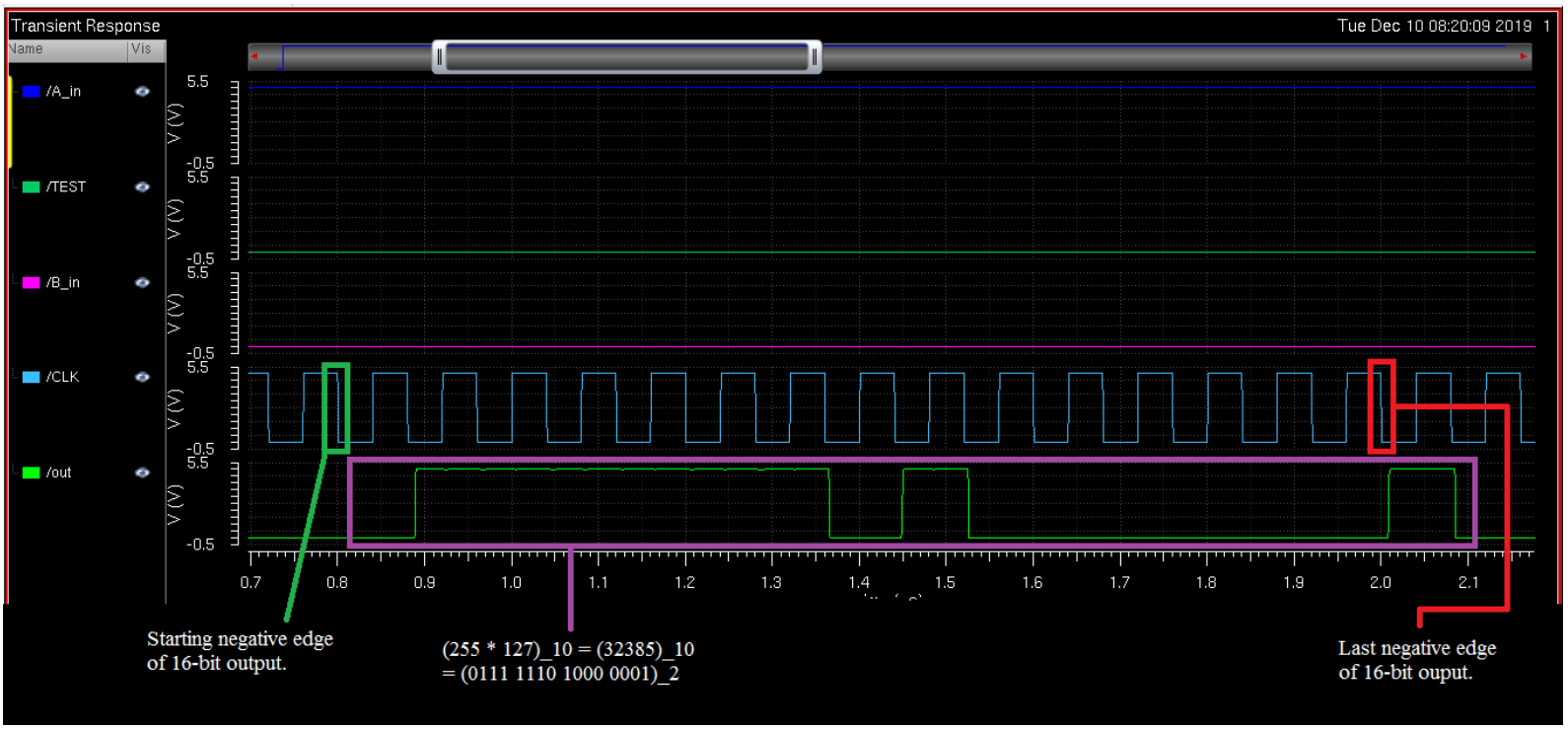



II. Full Adder with AND gate:

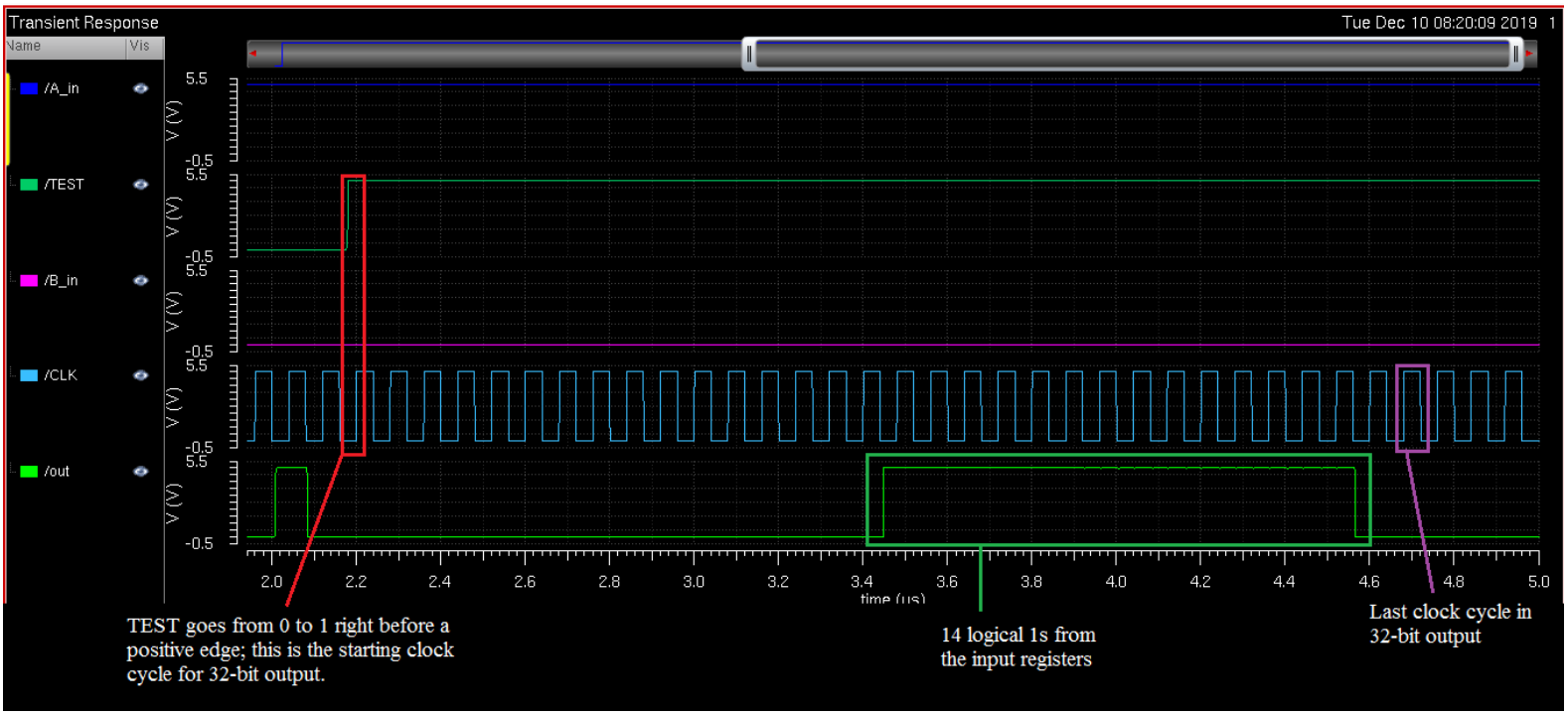


III. Registers (Test Mode):



14 logical 1s come out, but the input registers had 15.



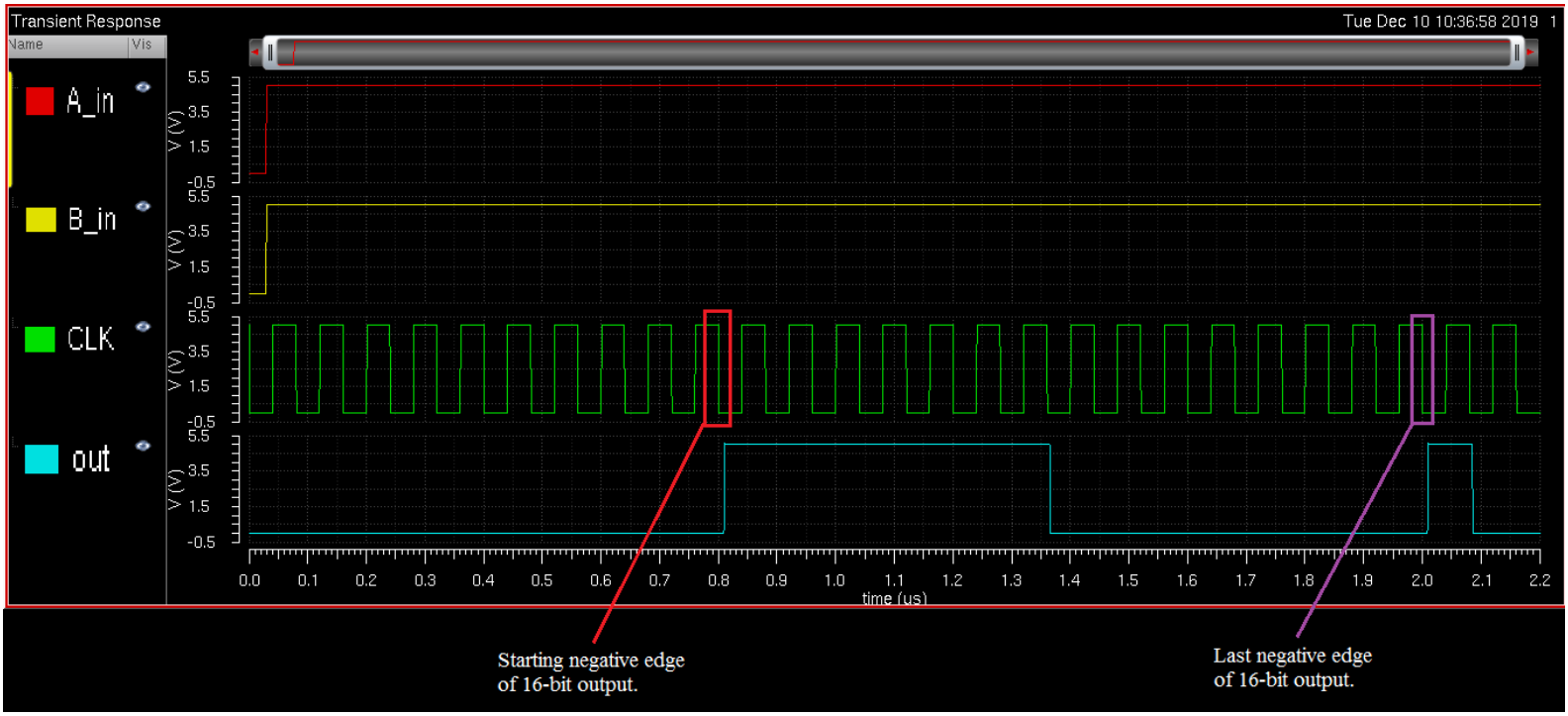
IV. Ring Oscillator (If used):

(b) (20 pts) The final multiplier:

**Starting edge of output is at 0.8 µs, or 800 ns and last edge is at 2 µs or 2000 ns.**

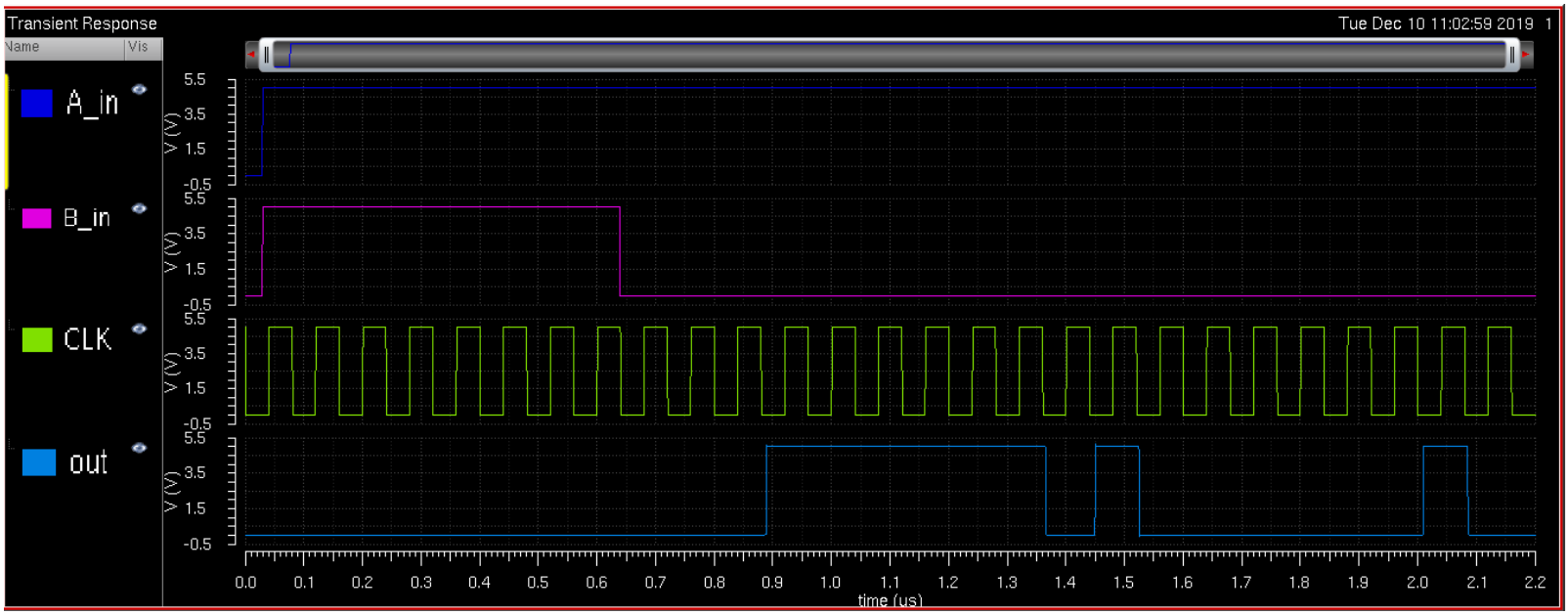
$A = (1111\ 1111)_2 = (255)_{10}, B = (1111\ 1111)_2 = (255)_{10}$

$P = (1111\ 1110\ 0000\ 0001)_2$



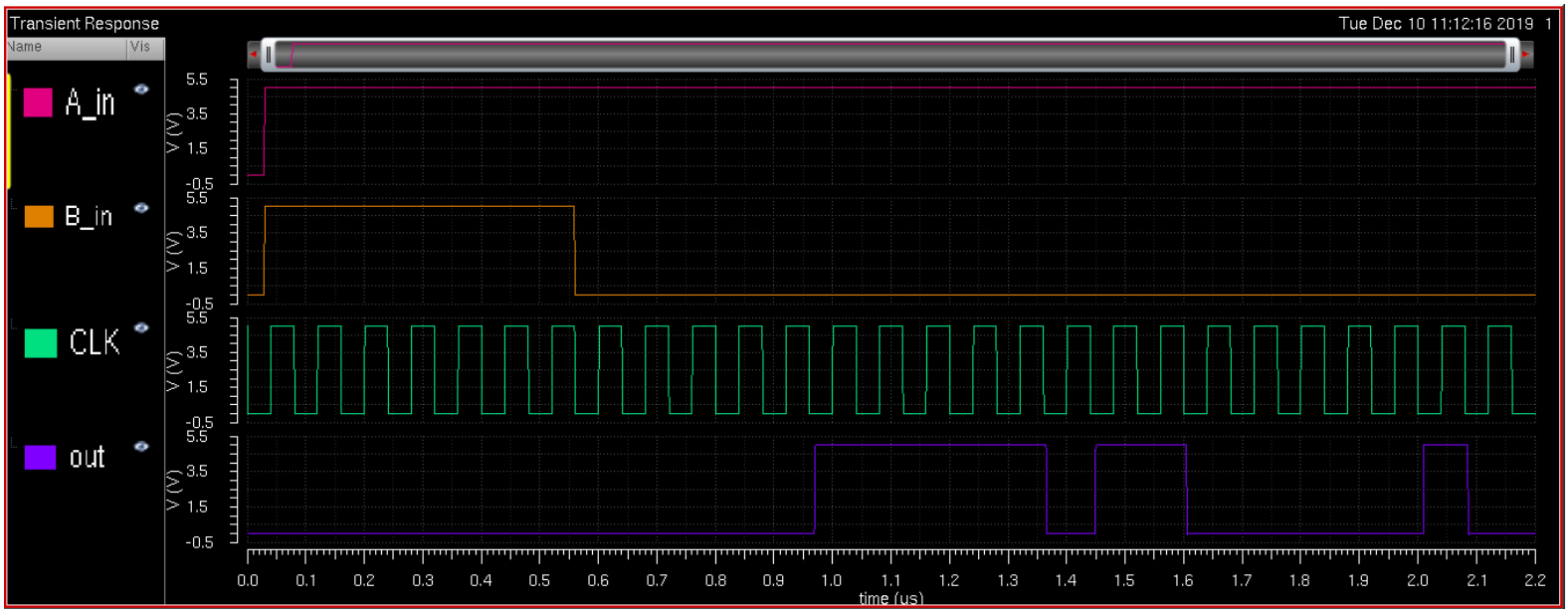
$A = (1111\ 1111)_2 = (255)_{10}, B = (0111\ 1111)_2 = (127)_{10}$

$P = (0111\ 1110\ 1000\ 0001)_2$



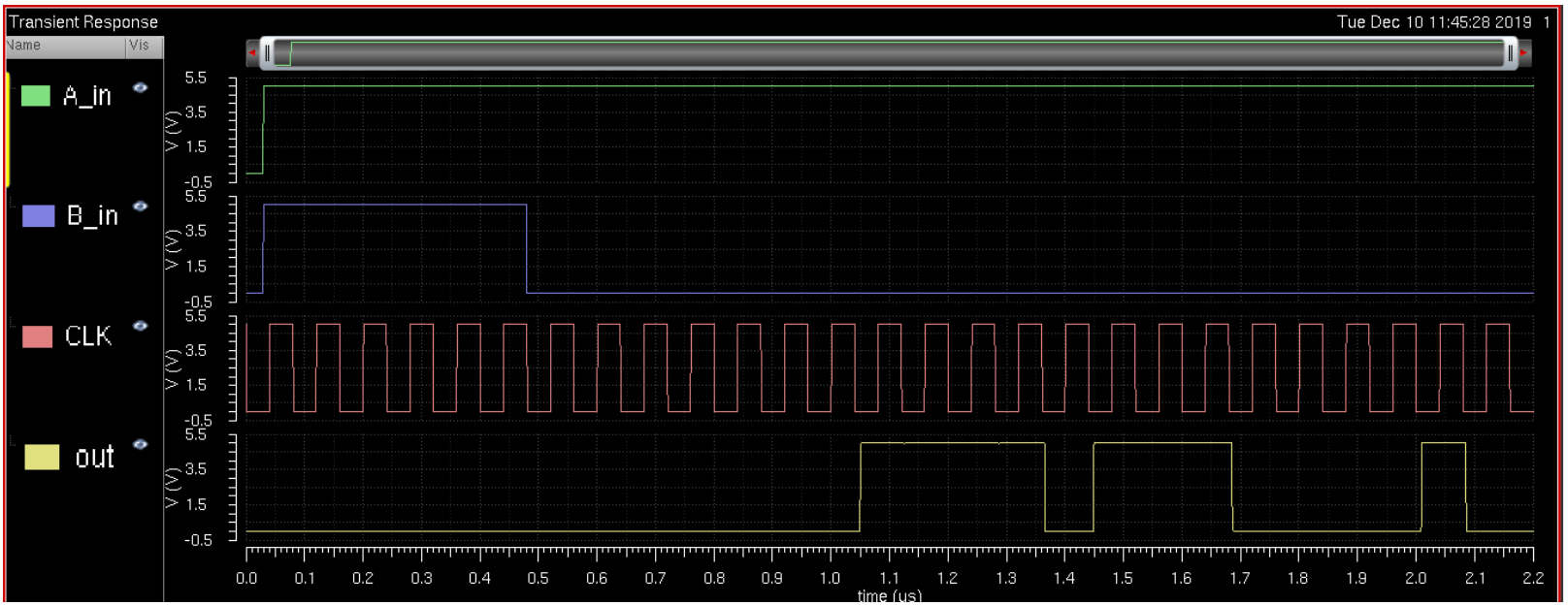
$A = (1111\ 1111)_2 = (255)_{10}, B = (0011\ 1111)_{\mathbf{2}} = (63)_{10}$

$P = (0011\ 1110\ 1100\ 0001)_2$



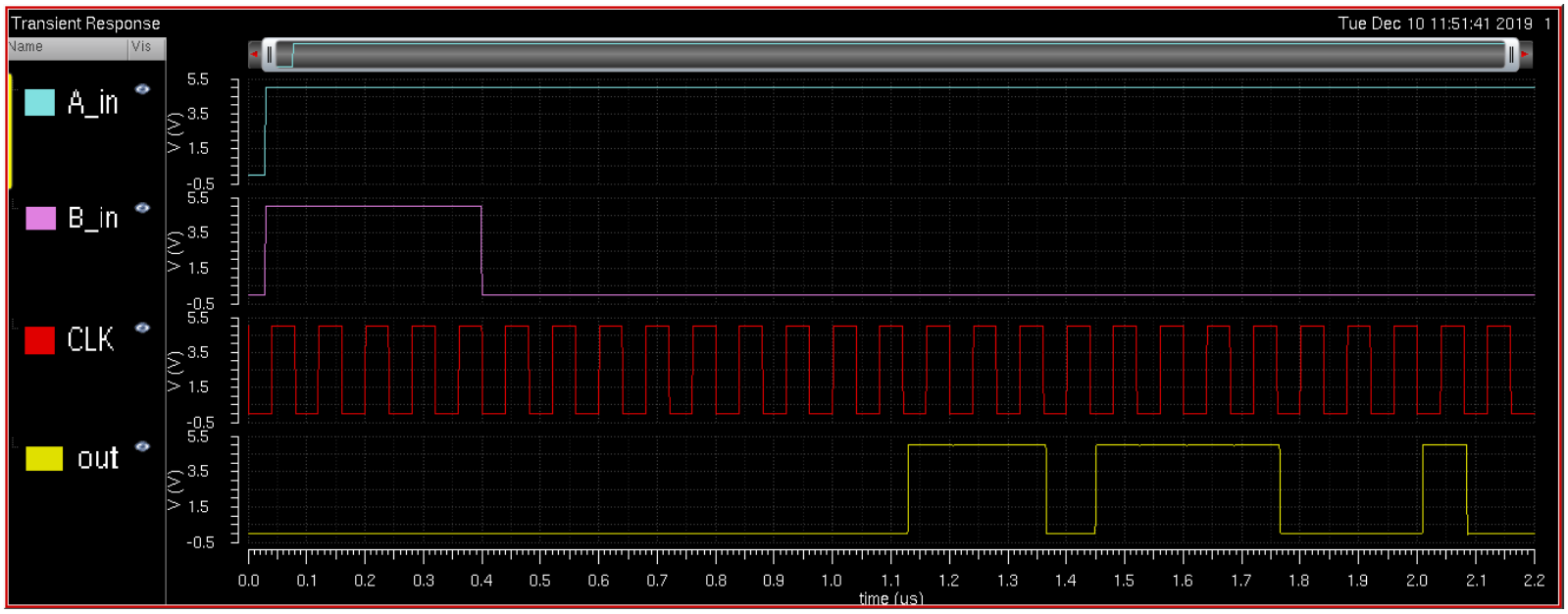
$A = (1111\ 1111)_2 = (255)_{10}, B = (0001\ 1111)_{\mathbf{2}} = (31)_{10}$

$P = (0001\ 1110\ 1110\ 0001)_2$



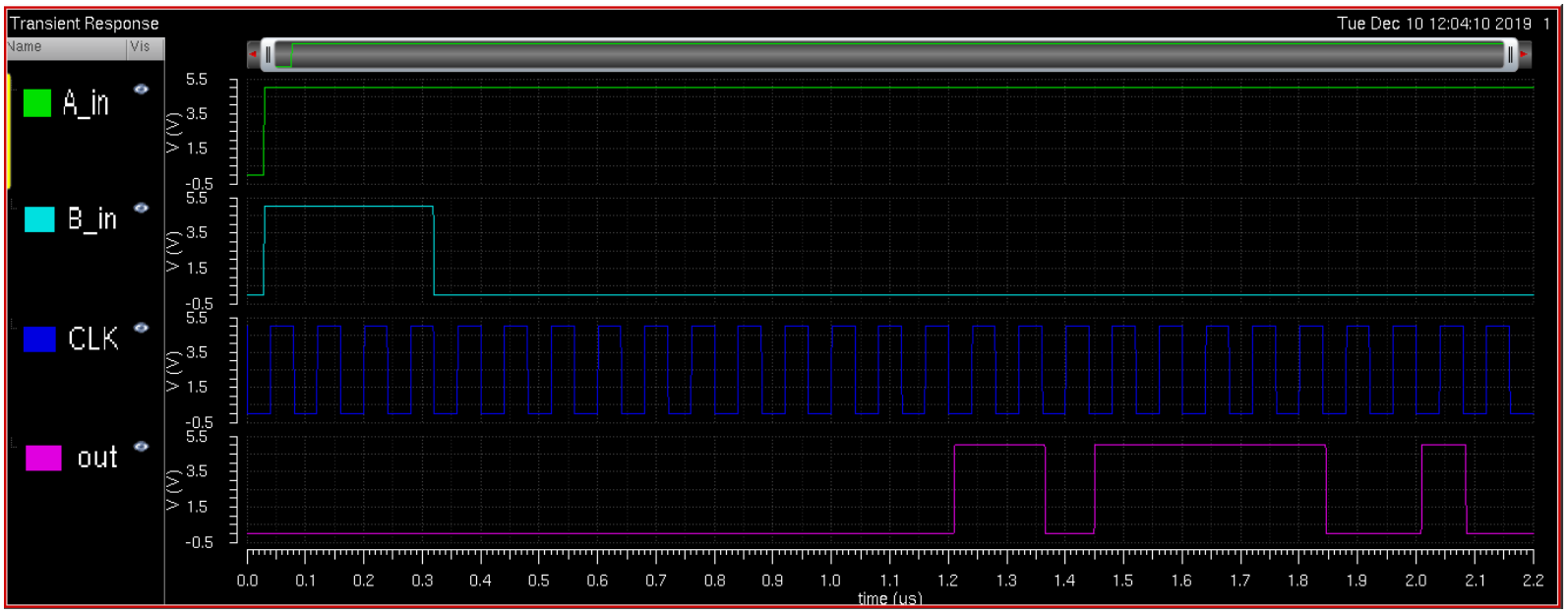
$A = (1111\ 1111)_2 = (255)_{10}, B = (0000\ 1111)_{\mathbf{2}} = (15)_{10}$

$P = (0000\ 1110\ 1111\ 0001)_2$



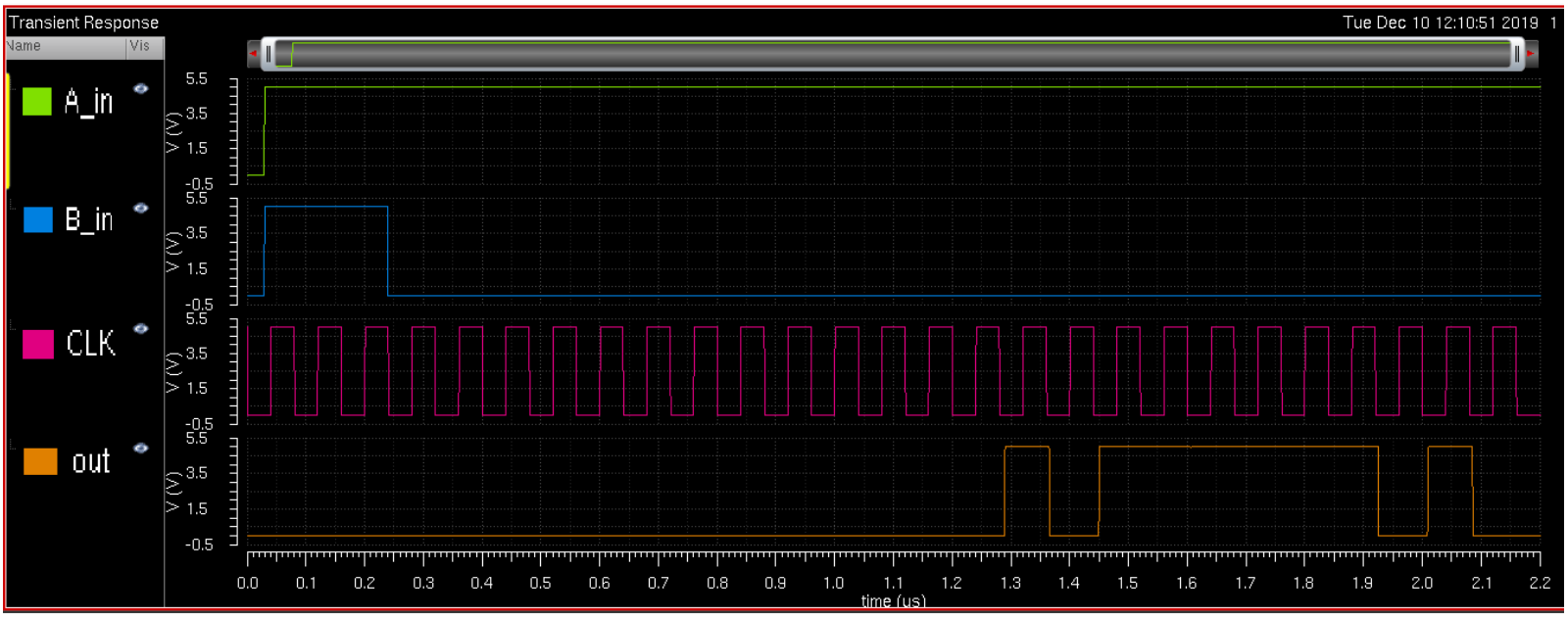
$A = (1111\ 1111)_2 = (255)_{10}, B = (0000\ 0111)_{\mathbf{2}} = (7)_{10}$

$P = (0000\ 0110\ 1111\ 1001)_2$



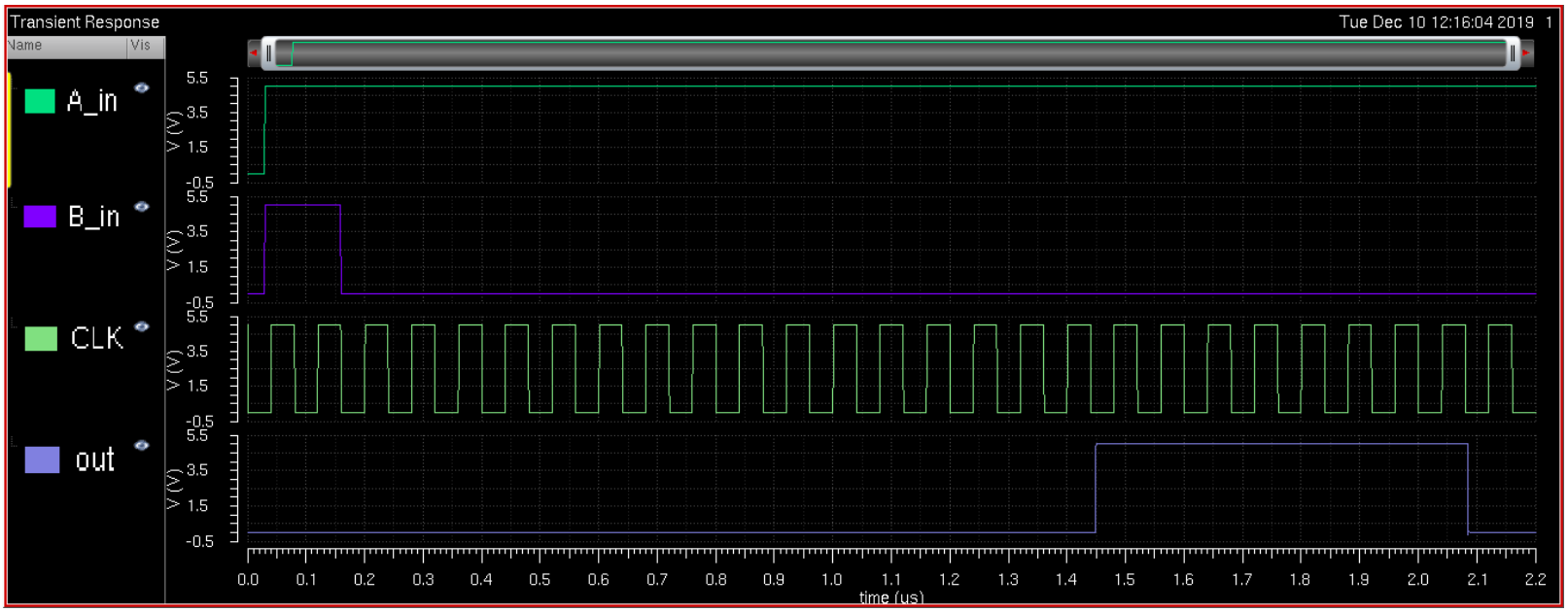
$A = (1111\ 1111)_2 = (255)_{10}, B = (0000\ 0011)_{\mathbf{2}} = (3)_{10}$

$P = (0000\ 0010\ 1111\ 1101)_2$

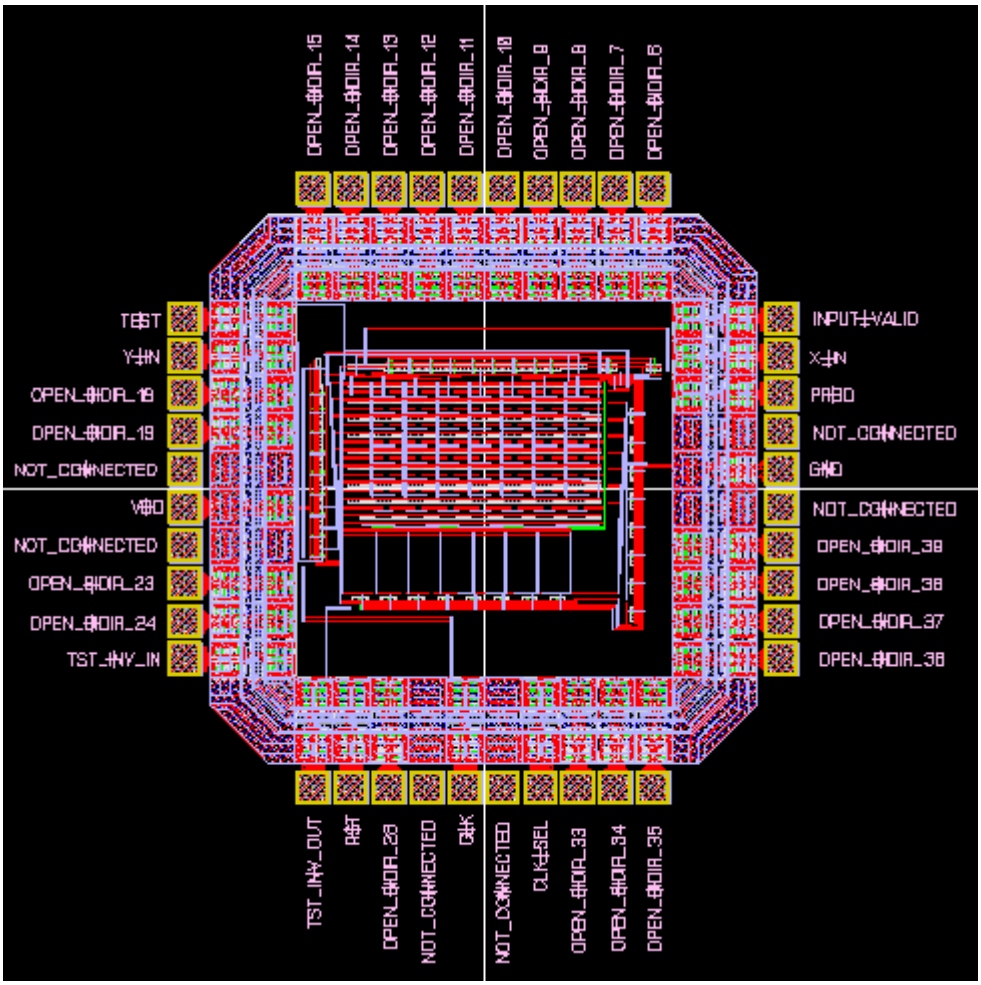


$A = (1111\ 1111)_2 = (255)_{10}, B = (0000\ 0001)_{\mathbf{2}} = (1)_{10}$

$P = (0000\ 0000\ 1111\ 1111)_2$



(4) **(10 pts.)** Layout of the final design (with padframe):



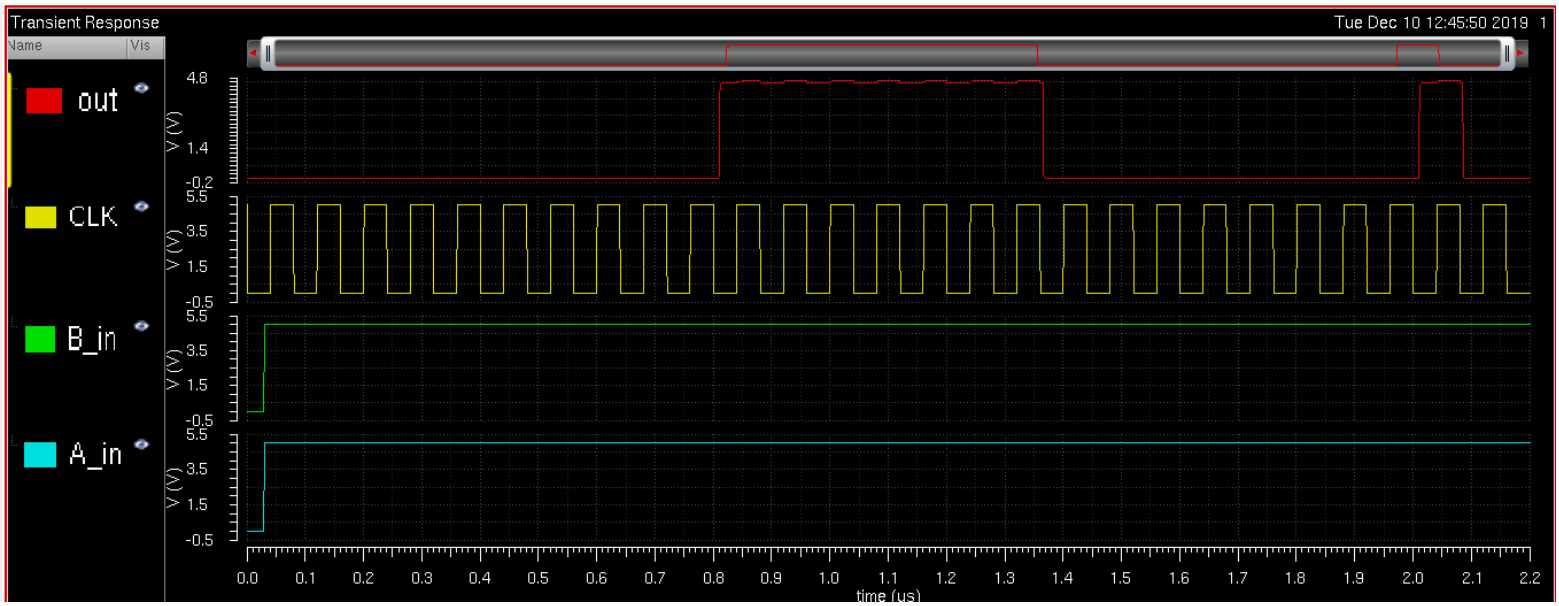
(5) **(20 pts)** Simulation waveforms for the final design (with padframe):

**Starting edge of output is at 0.8 µs, or 800 ns and last edge is at 2 µs or 2000 ns.**

Logical 1 slightly degraded at output (by 0.4 to 0.6 V) after connecting to the padframe.

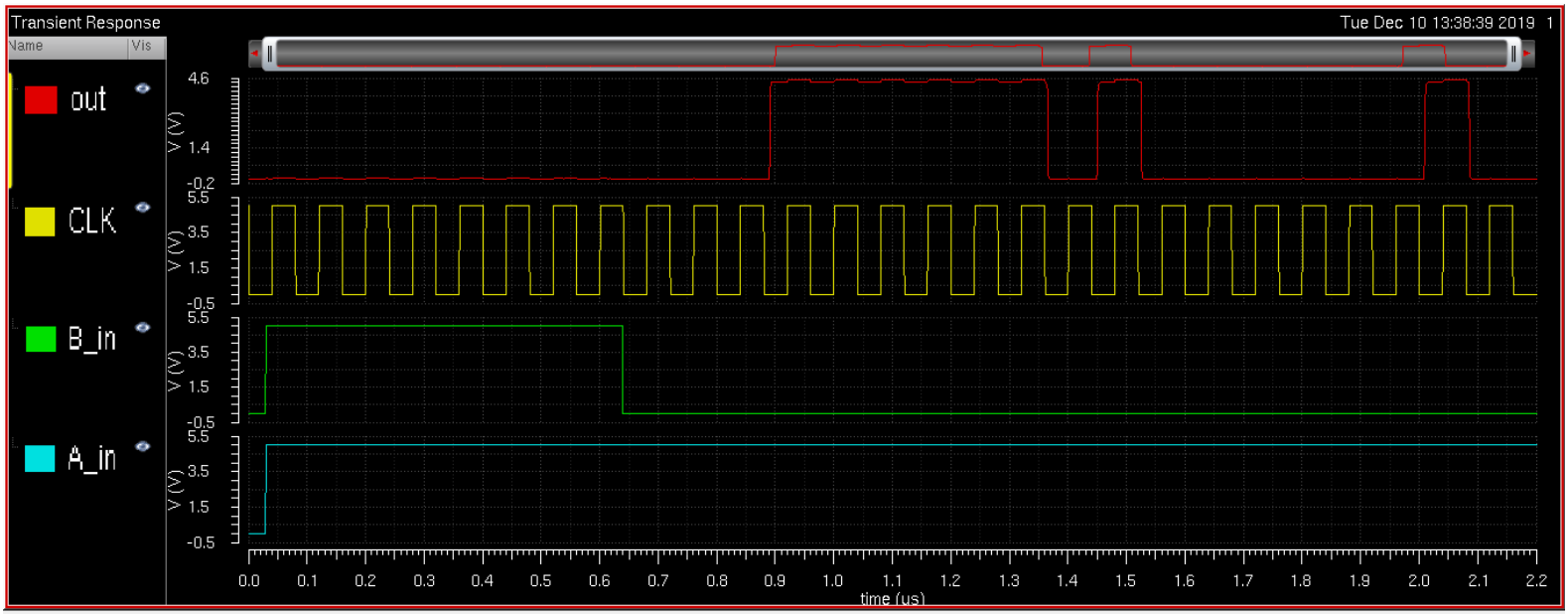
$A = (1111\ 1111)_2 = (255)_{10}, B = (1111\ 1111)_2 = (255)_{10}$

$P = (1111\ 1110\ 0000\ 0001)_2$



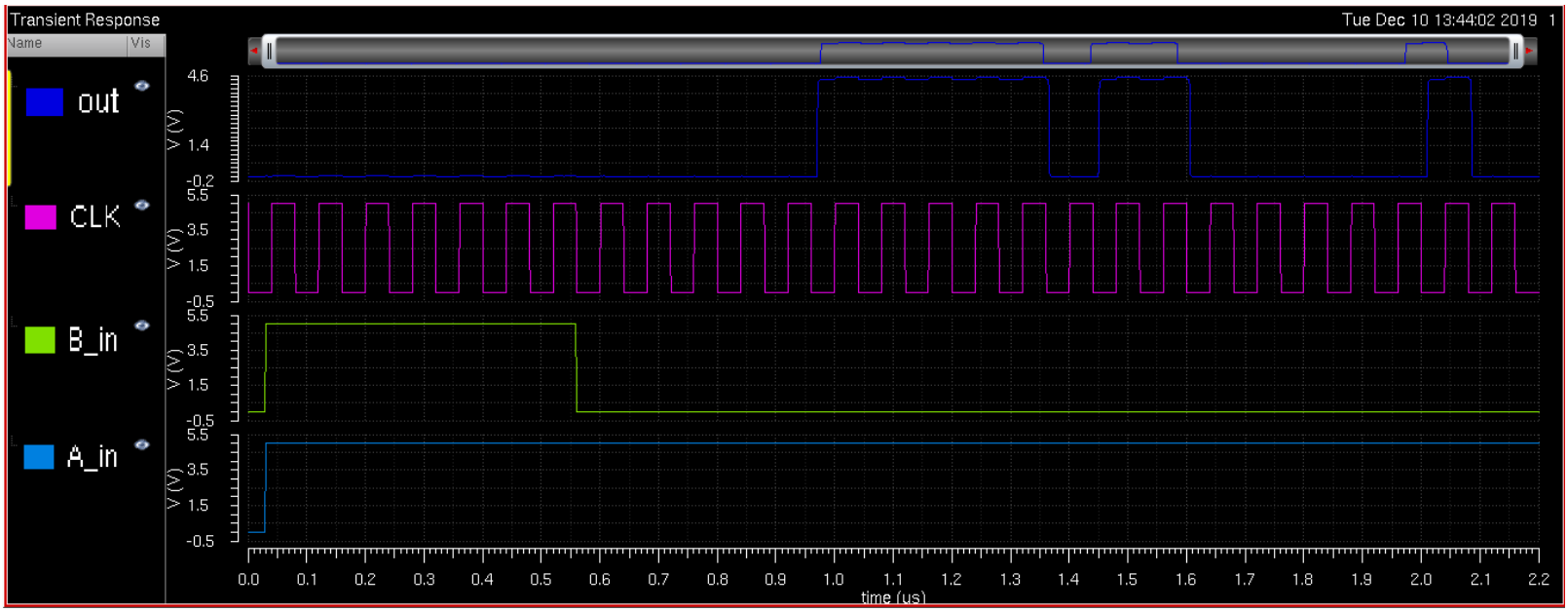

$A = (1111\ 1111)_2 = (255)_{10}, B = (0111\ 1111)_2 = (127)_{10}$

$P = (0111\ 1110\ 1000\ 0001)_2$



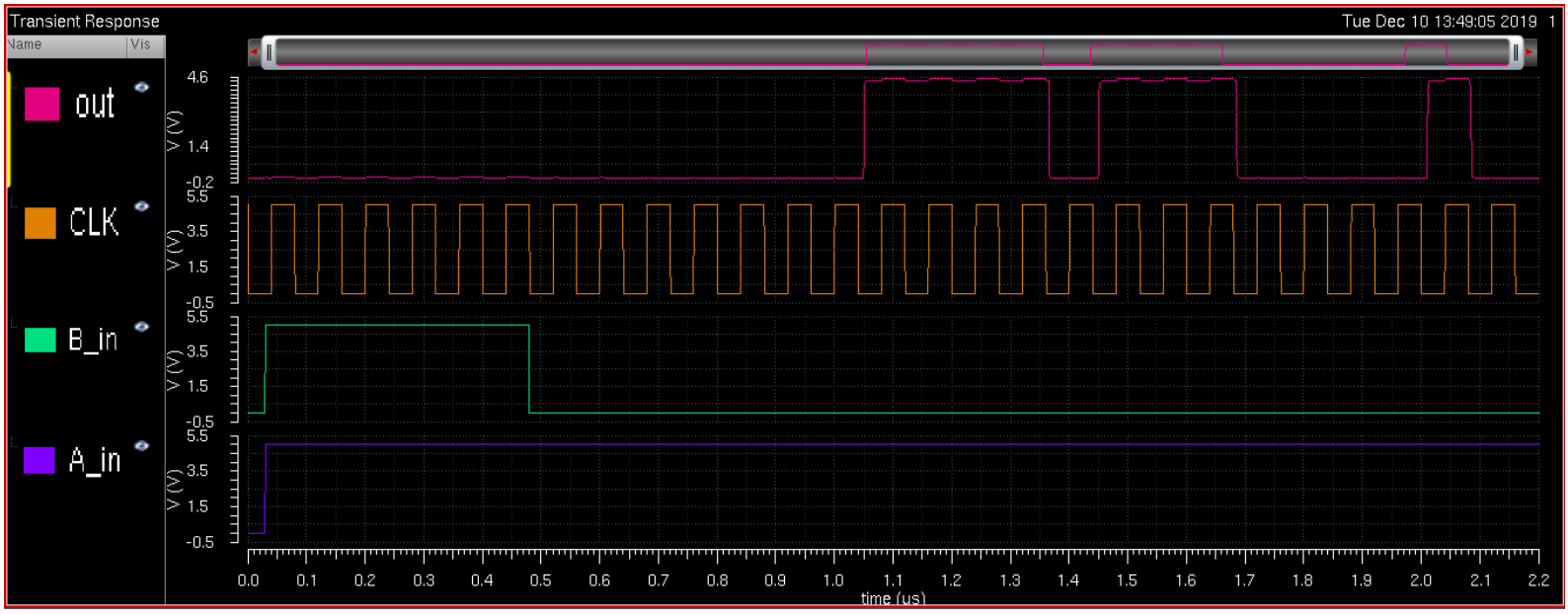
$A = (1111\ 1111)_2 = (255)_{10}, B = (0011\ 1111)_{\mathbf{2}} = (63)_{10}$

$P = (0011\ 1110\ 1100\ 0001)_2$



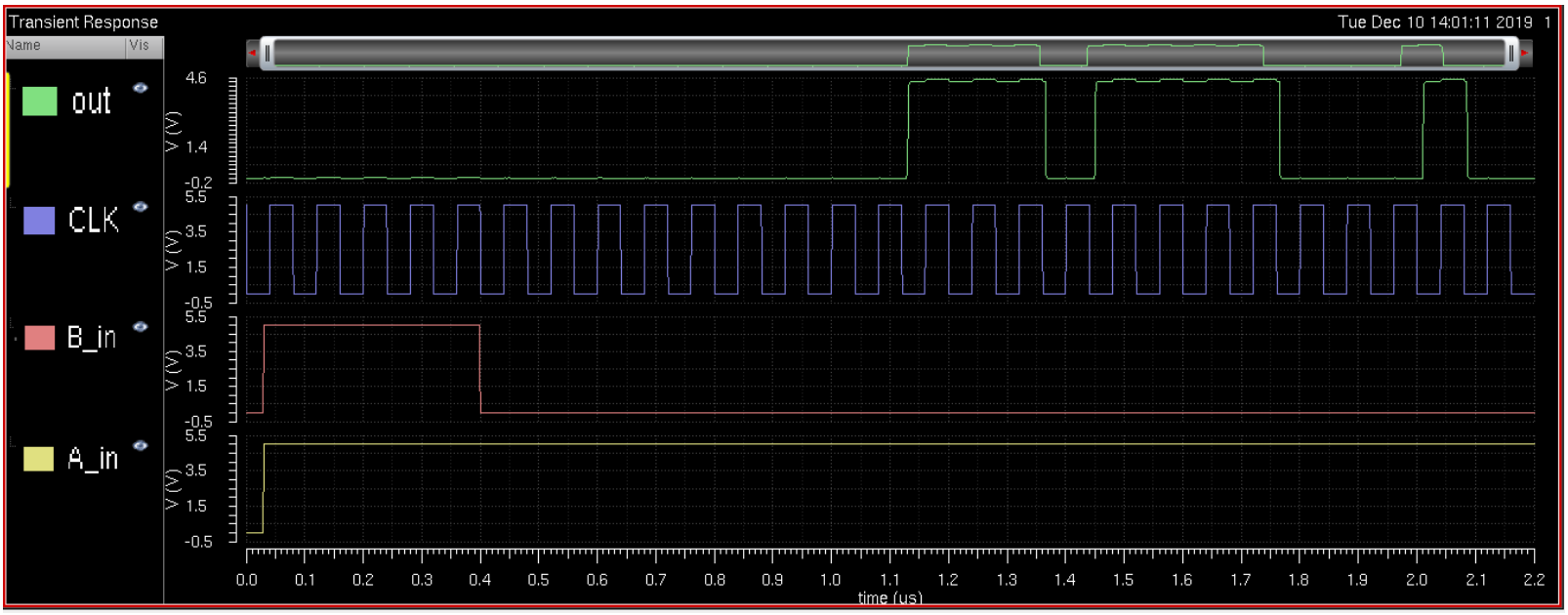

$A = (1111\ 1111)_2 = (255)_{10}, B = (0001\ 1111)_{\mathbf{2}} = (31)_{10}$

$P = (0001\ 1110\ 1110\ 0001)_2$



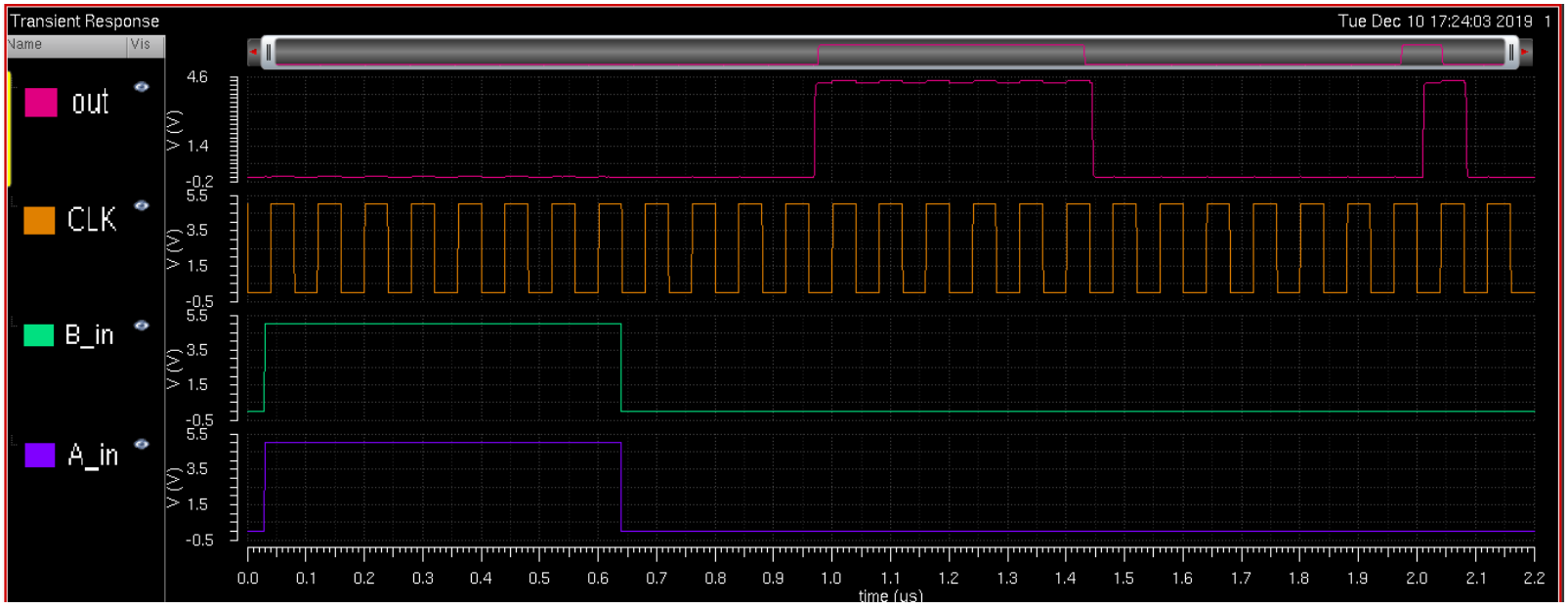
$A = (1111\ 1111)_2 = (255)_{10}, B = (0000\ 1111)_{\mathbf{2}} = (15)_{10}$

$P = (0000\ 1110\ 1111\ 0001)_2$



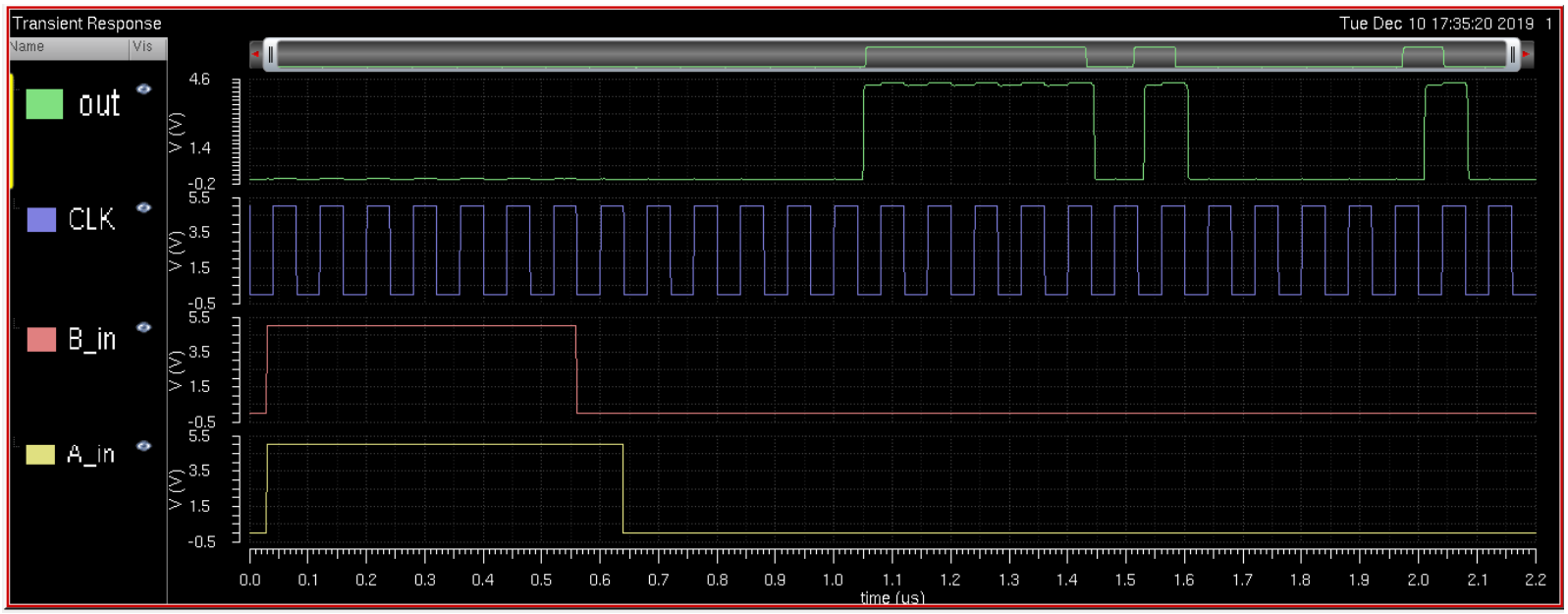

$A = (0111\ 1111)_2 = (127)_{10}, B = (0111\ 1111)_{\mathbf{2}} = (127)_{10}$

$P = (0011\ 1111\ 0000\ 0001)_2$



$A = (0111\ 1111)_2 = (127)_{10}, B = (0011\ 1111)_{\mathbf{2}} = (63)_{10}$

$P = (0001\ 1111\ 0100\ 0001)_2$




$A = (0011\ 1111)_2 = (63)_{10}, B = (0011\ 1111)_{\mathbf{2}} = (63)_{10}$

$P = (0000\ 1111\ 1000\ 0001)_2$