CDA 4213/CIS 6930 CMOS VLSI
Fall 2019

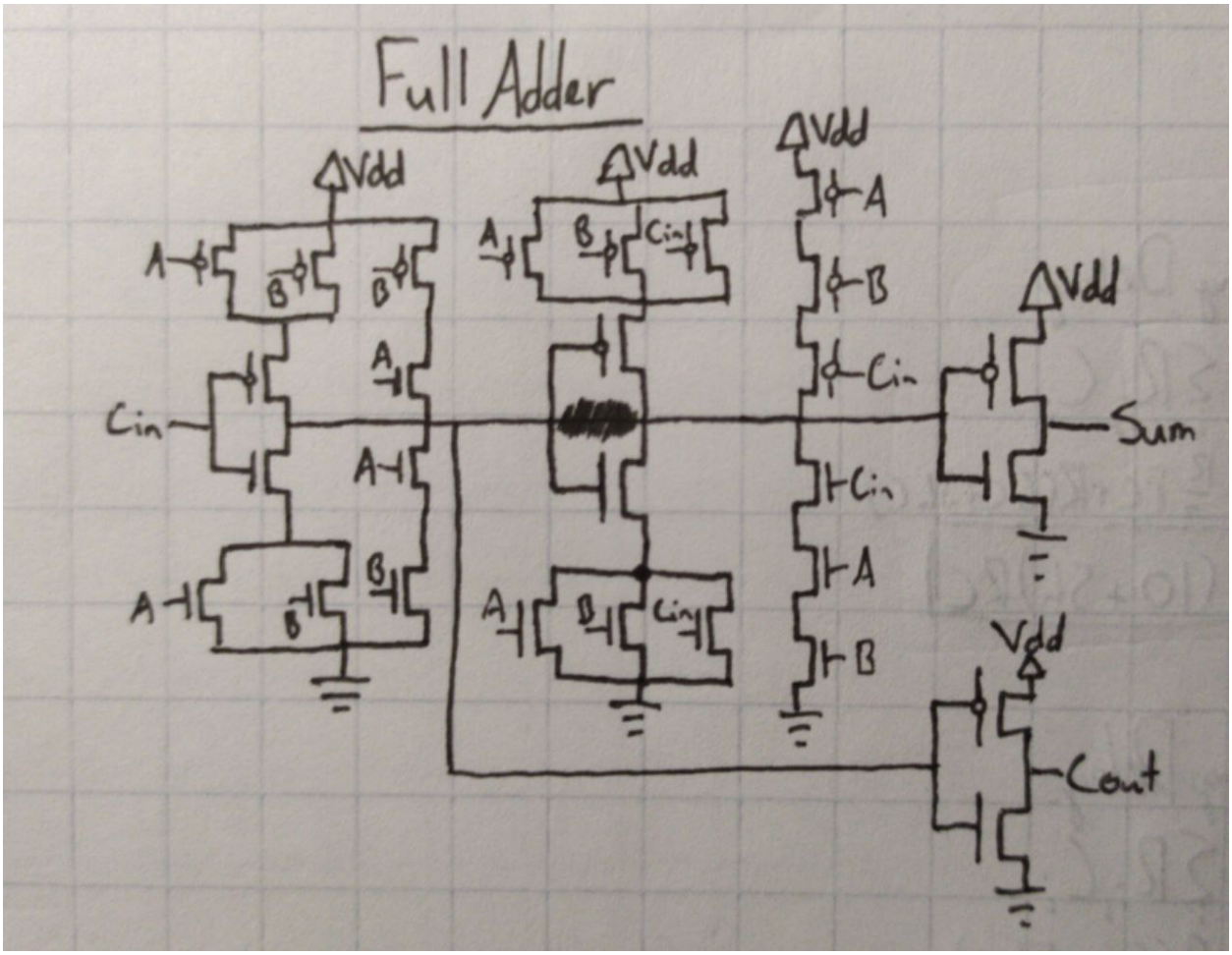
Final Project

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

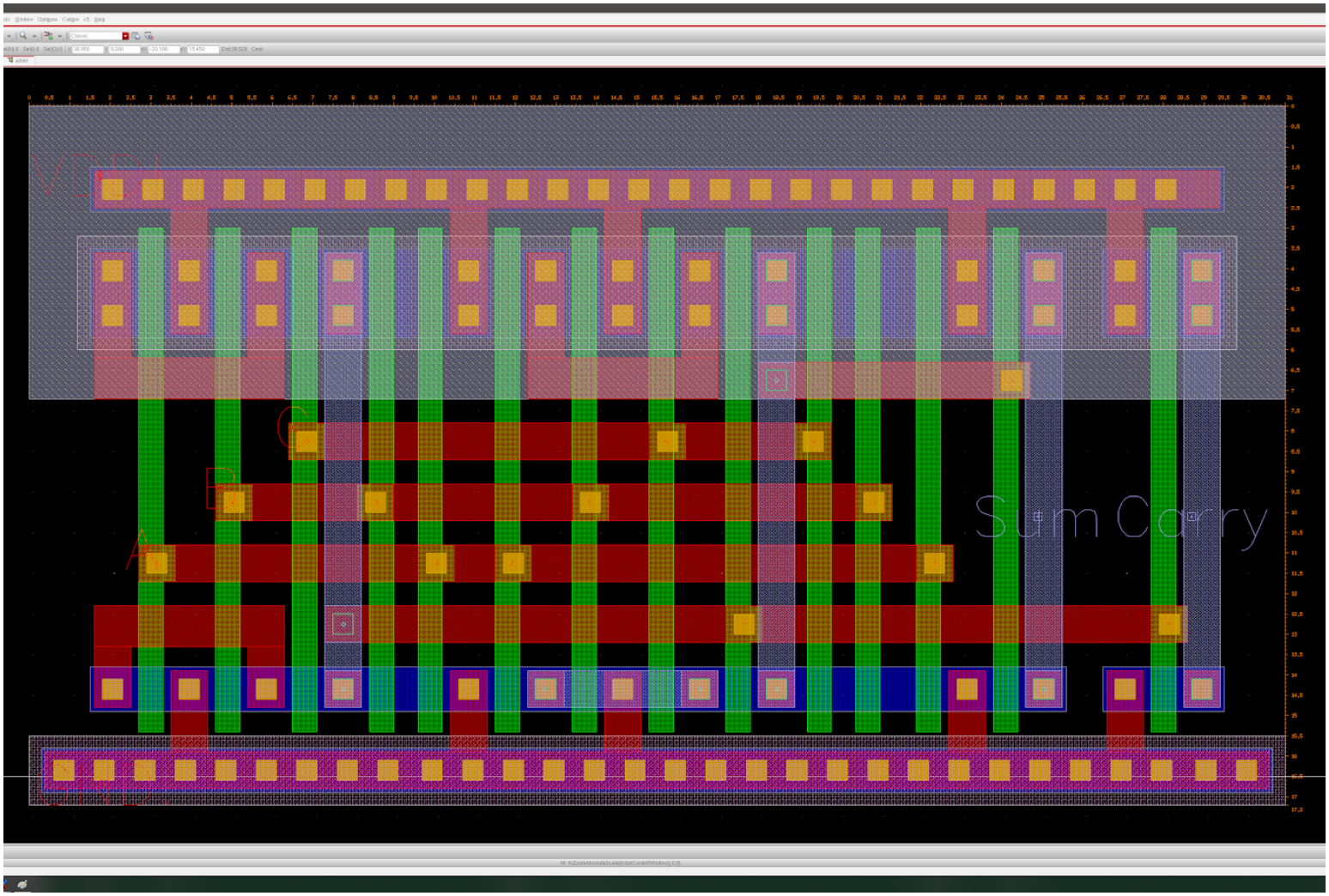
Today's Date:	12/8/2019
Your Team Name:	Trenislav Wu
Team Members:	Trent Callahan, Denislav Tsonev, Boyang Wu
Work Distribution	1) Trent Callahan - Created AND and MUX with Denislav. Debugged and tested 8x8 multiplier and shift registers. Drew gate-level designs for lab report and worked on the rest of the report. 2) Denislav Tsonev - Created AND and MUX with Trent. Debugged and tested 8x8 multiplier and shift registers. Designed a logic design for the test case. Worked on report. 3) Boyang Wu - Built 8x8 multipliers with existing gates and wired everything that was still not proper after the mosaic/copying. Debugged 8x8 multipliers.
No. of Hours Spent:	100+
Exercise Difficulty: (Easy, Average, Hard)	Average/Hard . The design was not hard with mosaics/copying but debugging wiring issues took a ton of time. Also due to how long it takes to run PEX (15 min w. padframe) /ADE (6-15 minutes/simulation) lots of time was spent waiting.
Any Feedback:	If the images are too small try zooming in.

1) **(10 pts)** Proposed Design – Bit slice design
   a) List all module bit-slices you have used for your design.
      i) Full Adder
      ii) AND-Adder
      iii) AND
      iv) 2-input MUX
      v) 8-bit Shift Register
      vi) 16-bit Shift Register
      vii) Ring Oscillator
   b) (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:

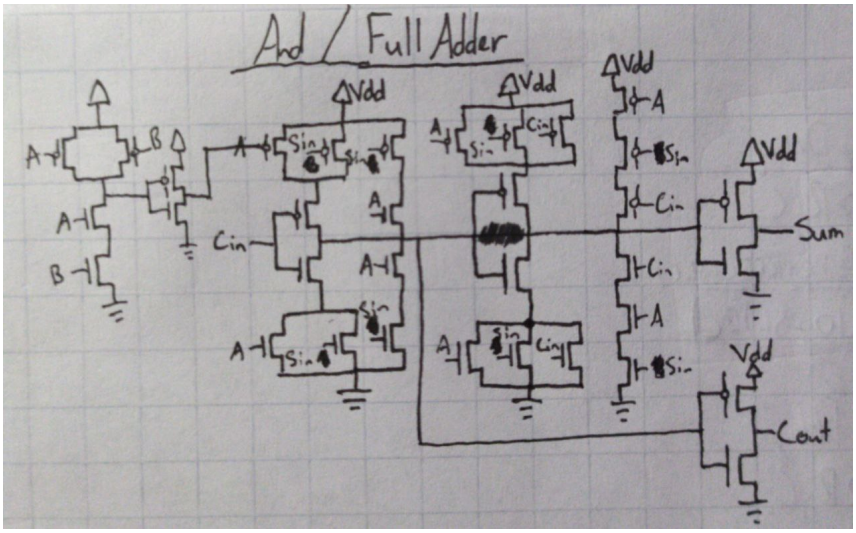


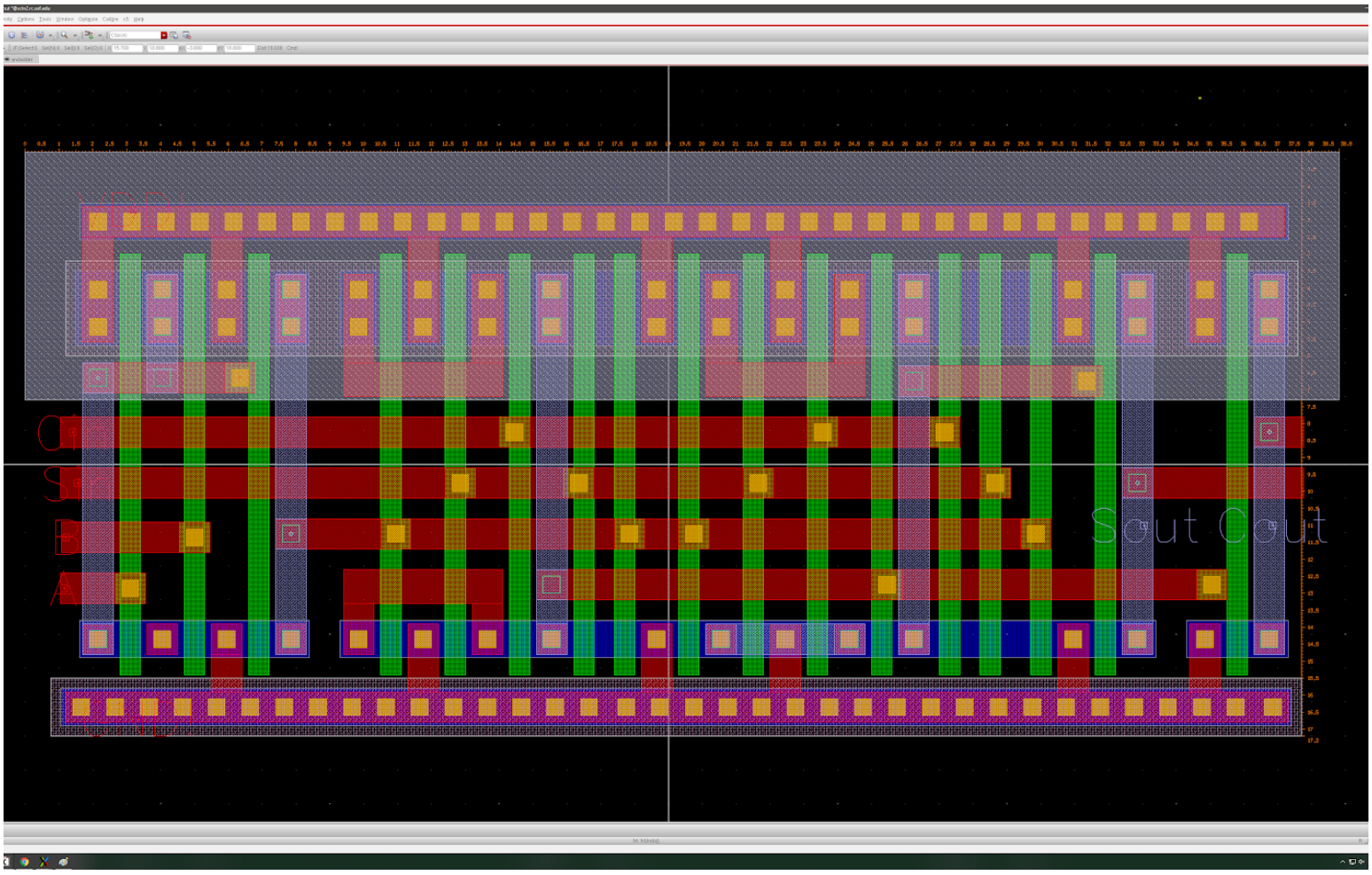
Full Adder:



17.2 x 31 = 533.2

II. Full Adder with AND gate:

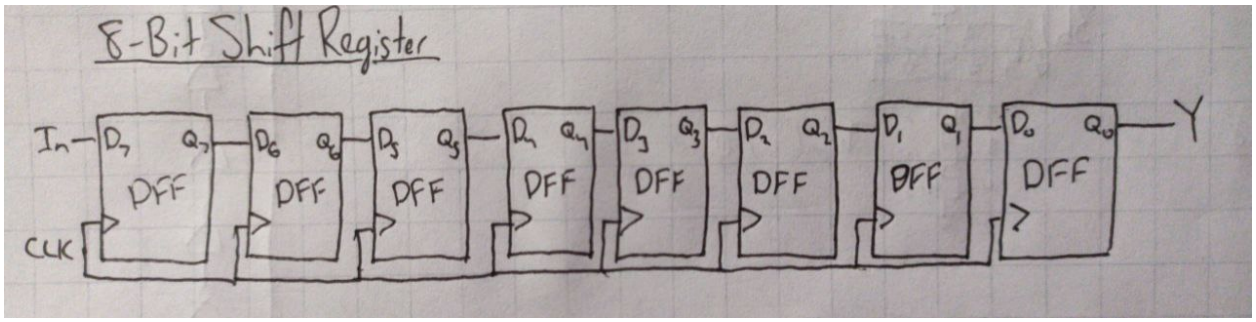


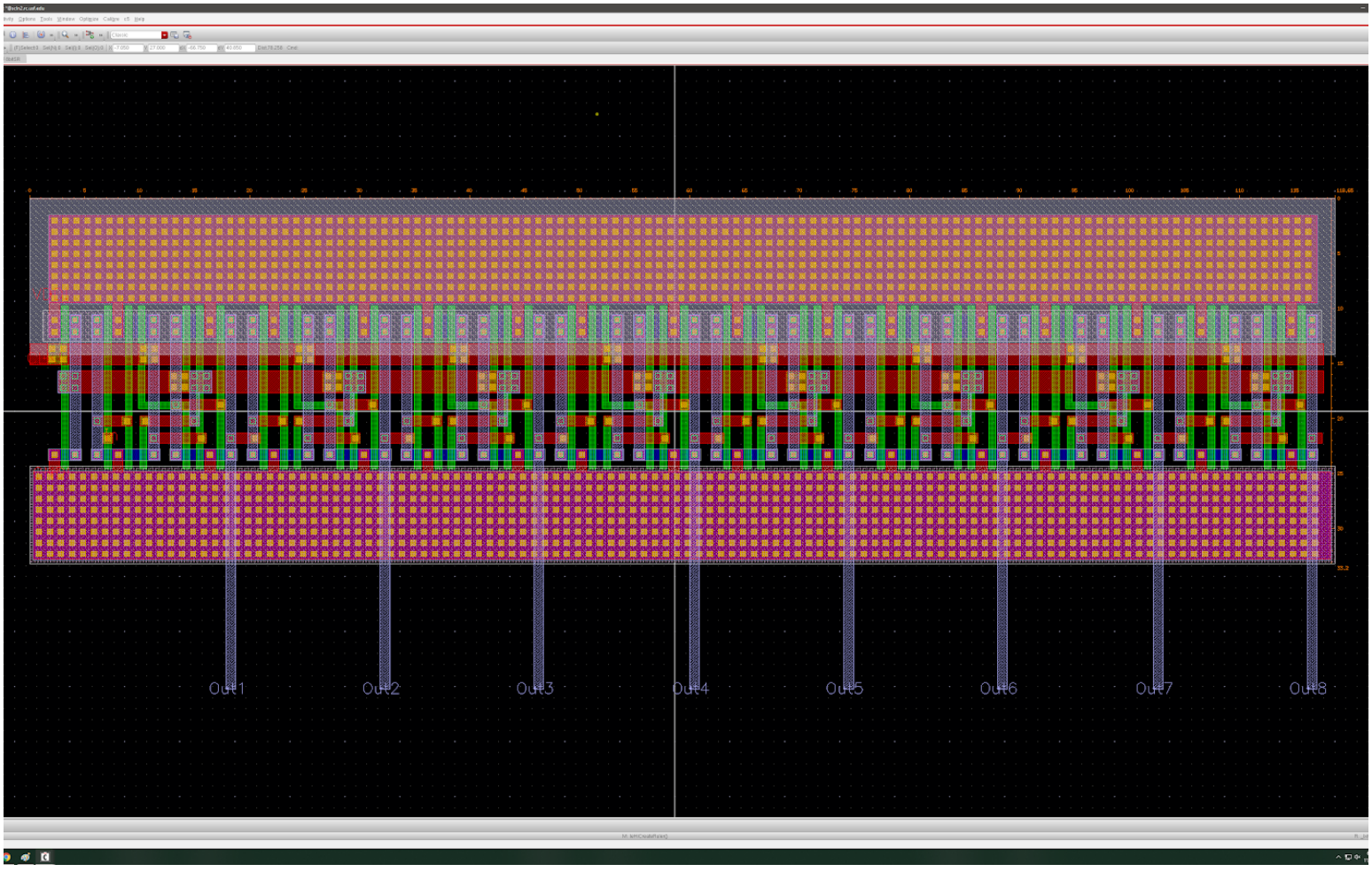


17.2 x 38.8 = 667.36

III. Registers (Inputs and Output):

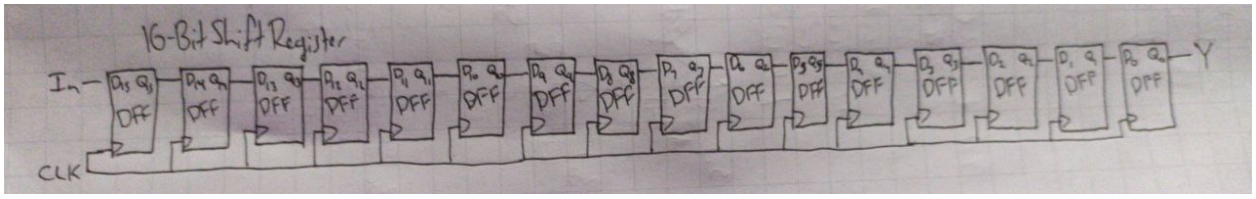
8-bit shift register (with wires for parallel out and serial out)

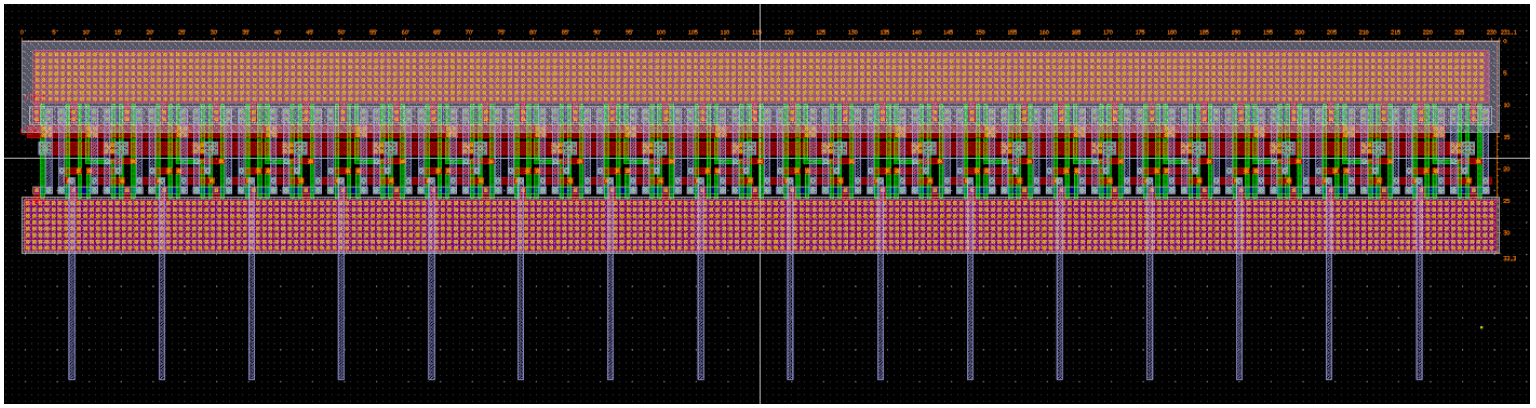




118.65 x 33.2 = 3939.18

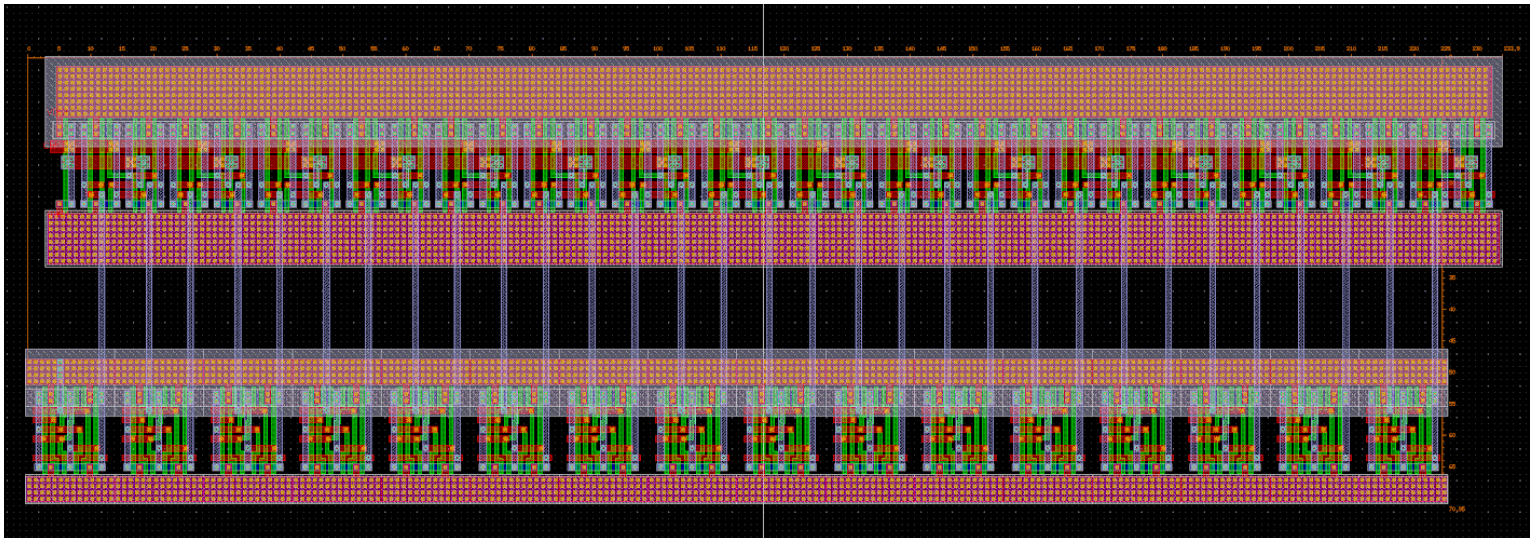
16-bit shift register (with wires for parallel out and serial out)





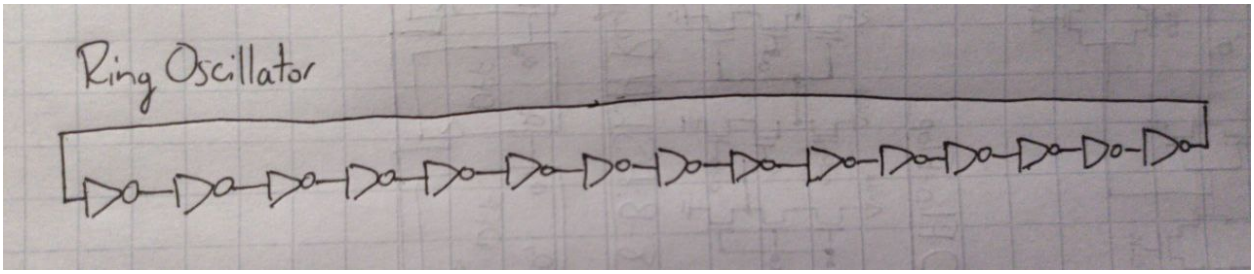
231.1 x 33.3 = 7695.63

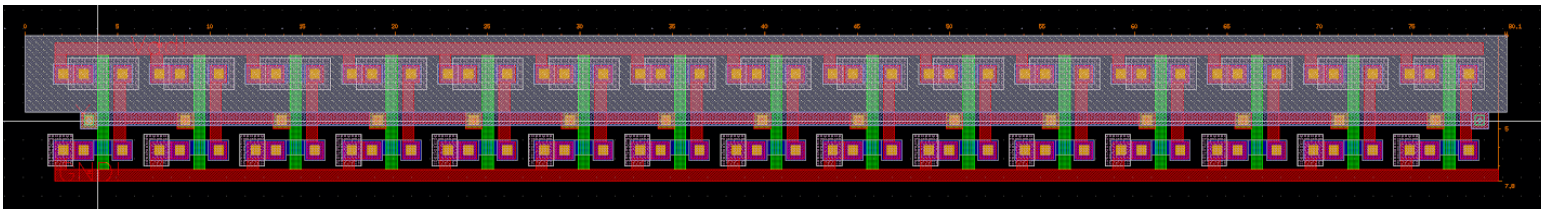
16-bit SR with 16 2 input multipliers (partial wiring for D1 in and Q out)



233.9 x 70.95 = 16595.205

IV. Ring Oscillator (If used):

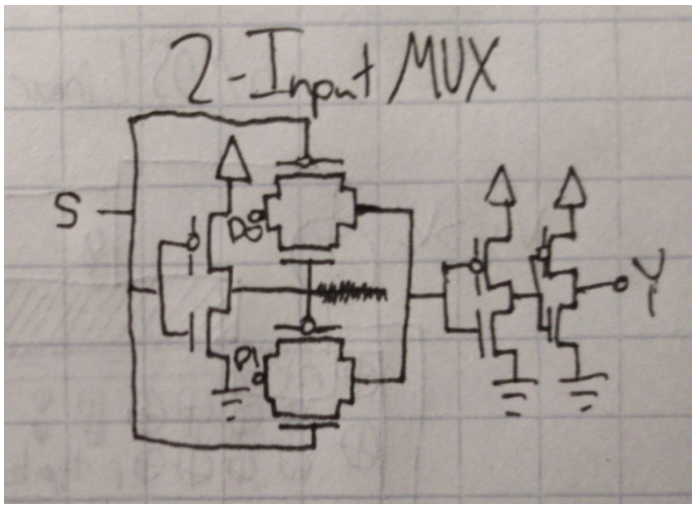


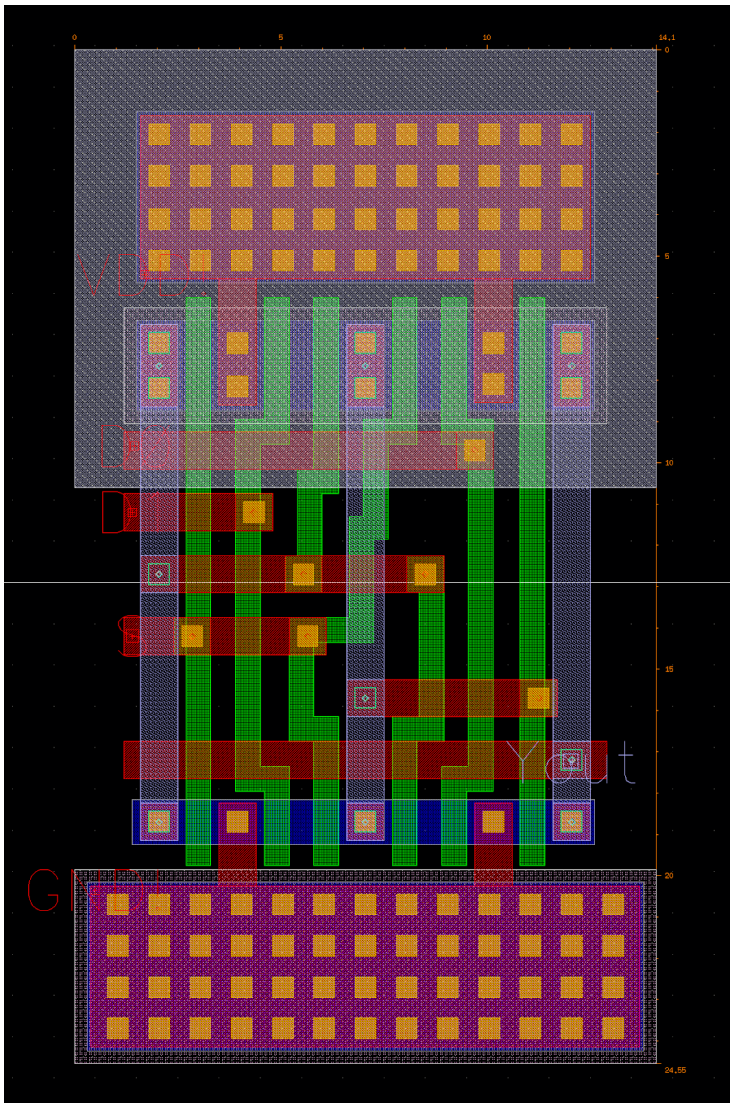


Currently works for a 7ns wavelength/delay. May need to be extended to around 15ns by doubling the number of inverters, but longer delay can be simulated by waiting 2 clock cycles instead of 1.

Not used in final design due to lack of time

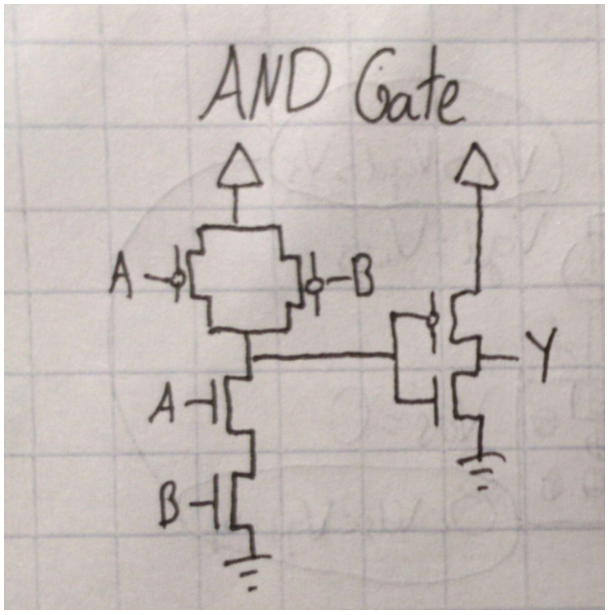
V. 2-input MUX

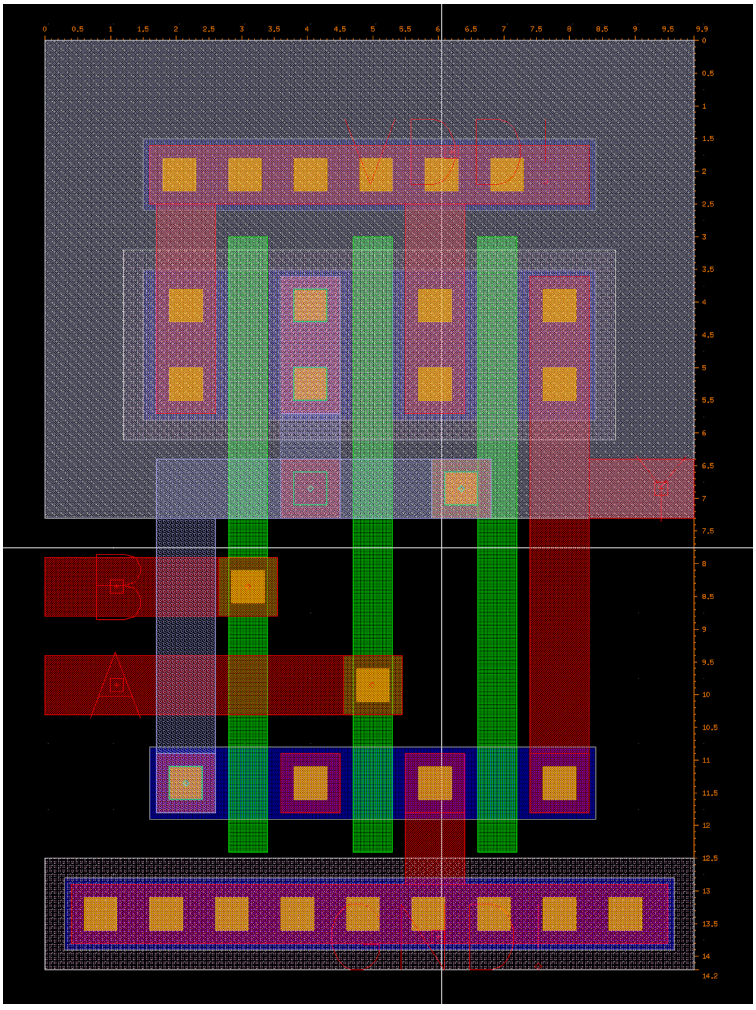




14.1*24.55 = 364.155

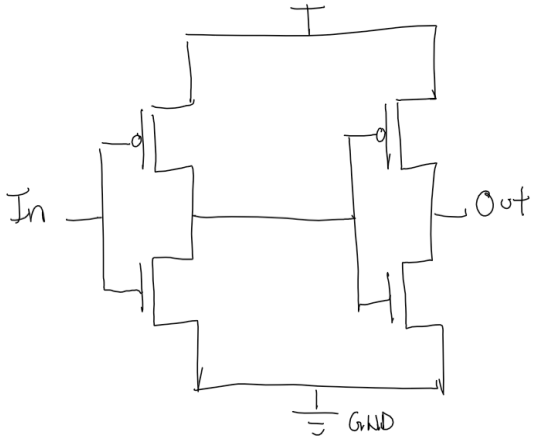
VI. AND

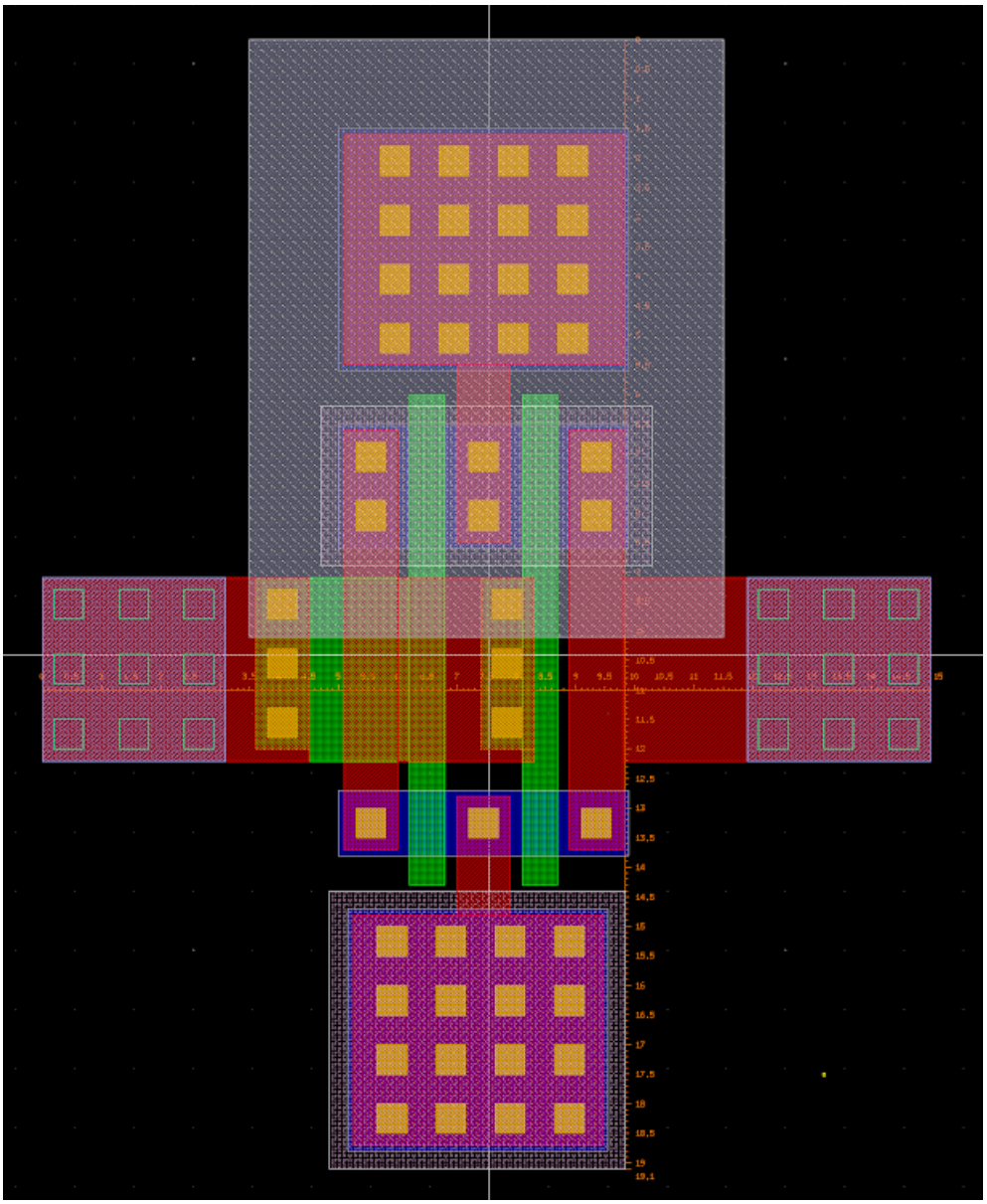




9.9 x 14.2 = 140.58

VII. Buffer - 15x19.1 = 286.5



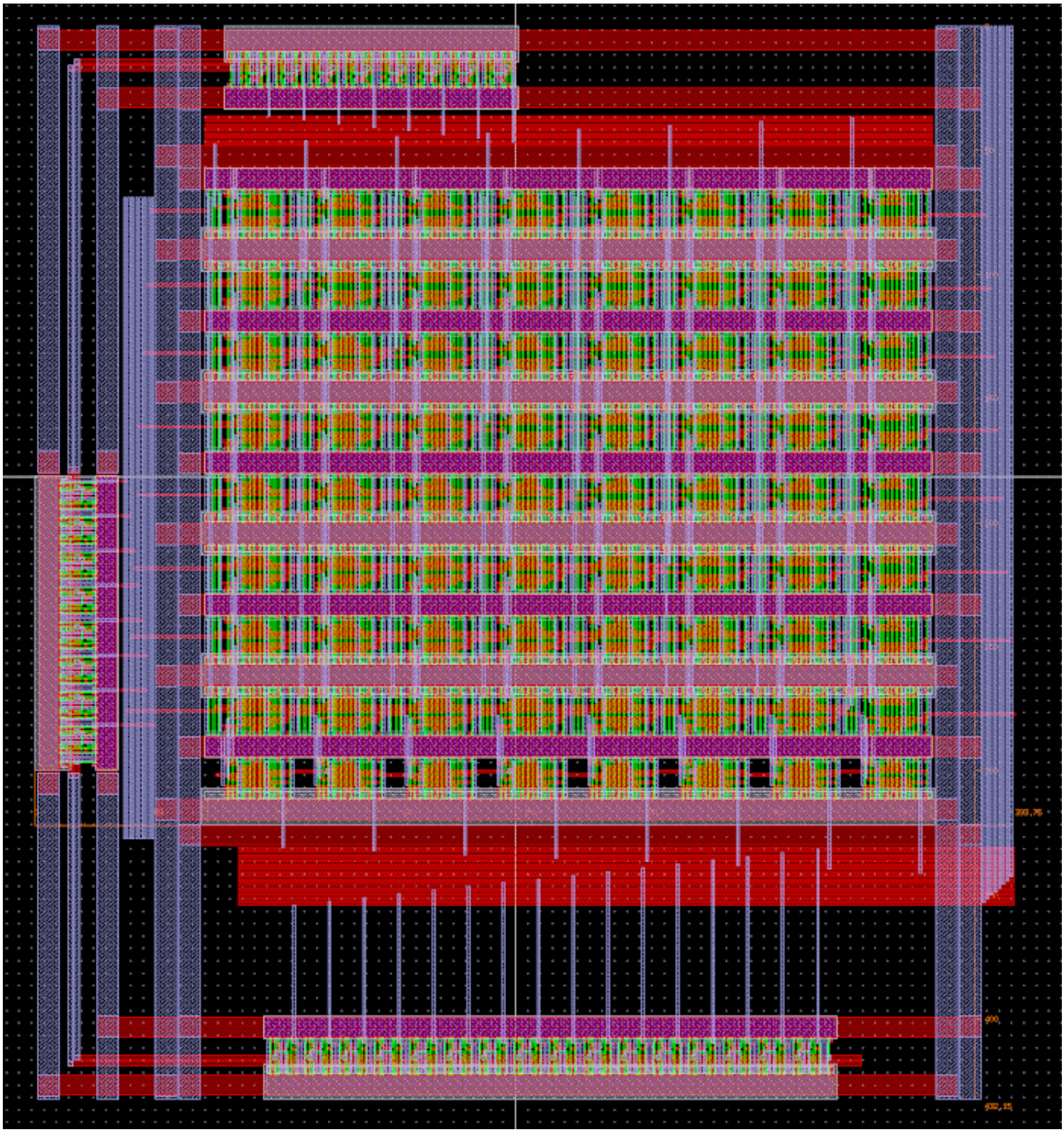


(2)**(15 pts.)** Show the layout of your multiplier with the registers (outside the padframe). Explain the design and functionality of your multiplier.

The design of our 8x8 multiplier is implemented through a ripple carry adder which connects full adders in series and has a carry bit that ripples and carries through the design. That way when you add multiple bits together we can keep track of the carry and if needed will carry over to the next ripple carry adder implemented. The 8-bit registers, which are our input, wait for 8 clock cycles until they are both full and represent an 8-bit number, which then is propagated into the combinational logic of the full adders connected in the middle. Those ripple-carry adders calculate the multiplication of both those numbers and then parallel out the number into the 16-bit shift register which is our output in the end. Finally, the parallel out is controlled by 16 multiplexers, which allow a selector bit to choose when we want to parallel in or serial out the results we have received.

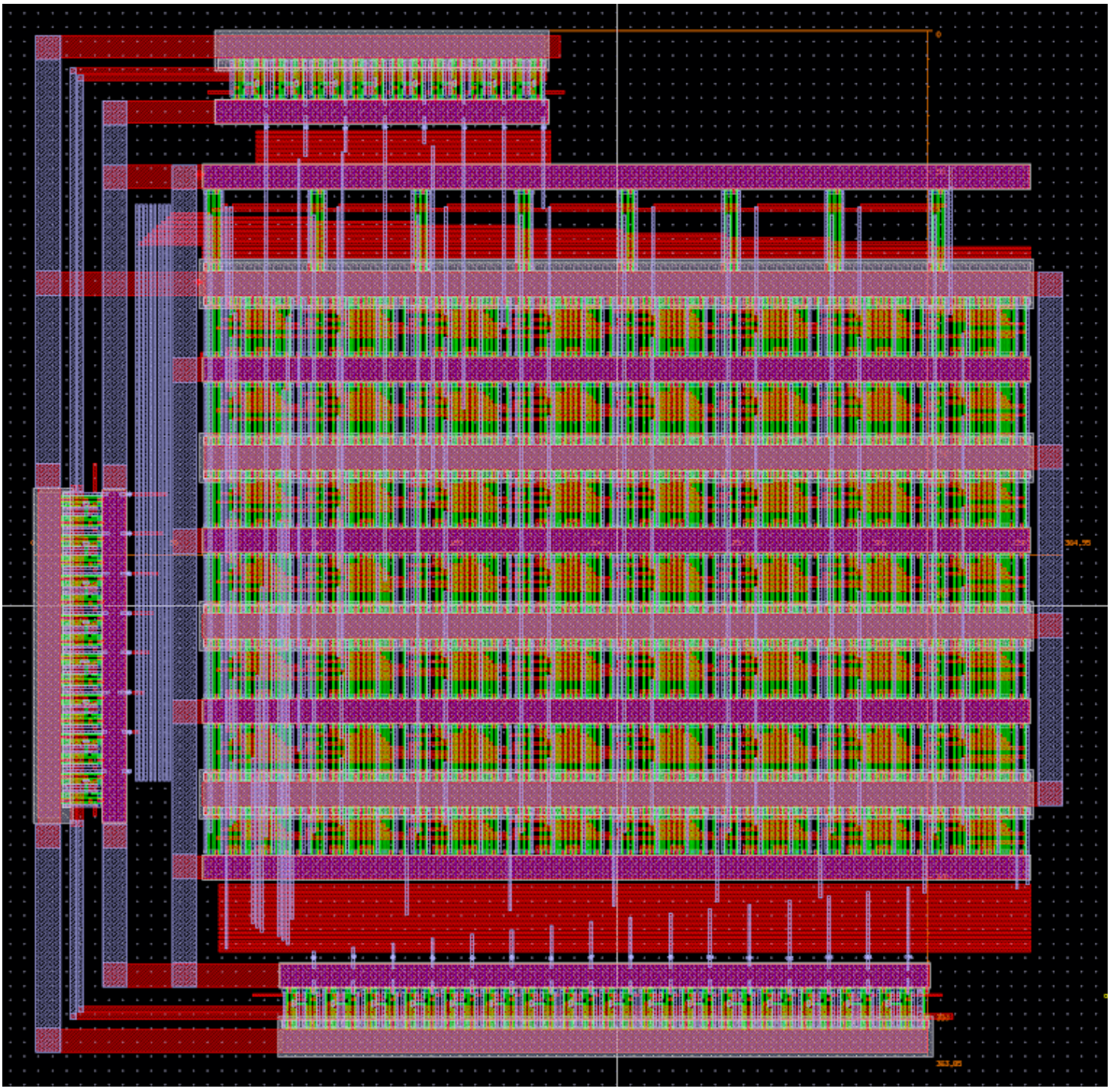
Layout is on the next page. 8-bit shift registers and 16-bit shift registers are in the layout but are currently not connected. They have wires going to (but are currently not connected to) buses in M1/M2 which are currently test inputs for A7-A0 and B7-B0 in place of the two 8-bit SRs and C15-C0 in place of the 16-bit SR.

New working design (8 rows of AND+Adders + 1 row of Adders going from right to left)



Old design (1 row of AND gates, 7 rows of AND+Adders)

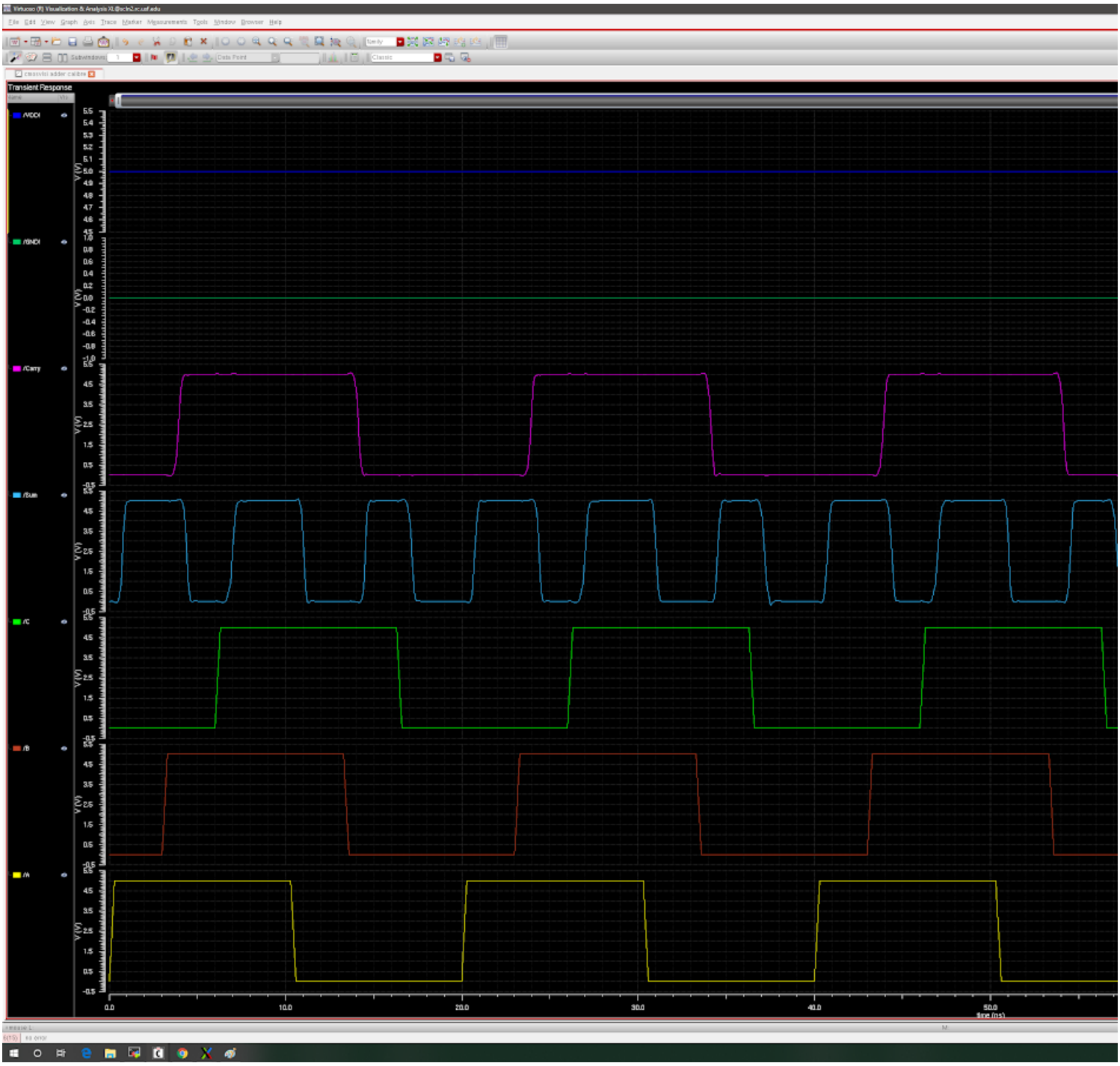
- Ended up not working at first since the 8-bit SRs were wired up to VDD and GND, but also the test lines. 8-bitSRs output 0 when the test lines (buses in M1/M2) input the needed values, which ended up causing issues for the final waveforms. Was mostly fixed so we have two working 8x8 multiplier designs



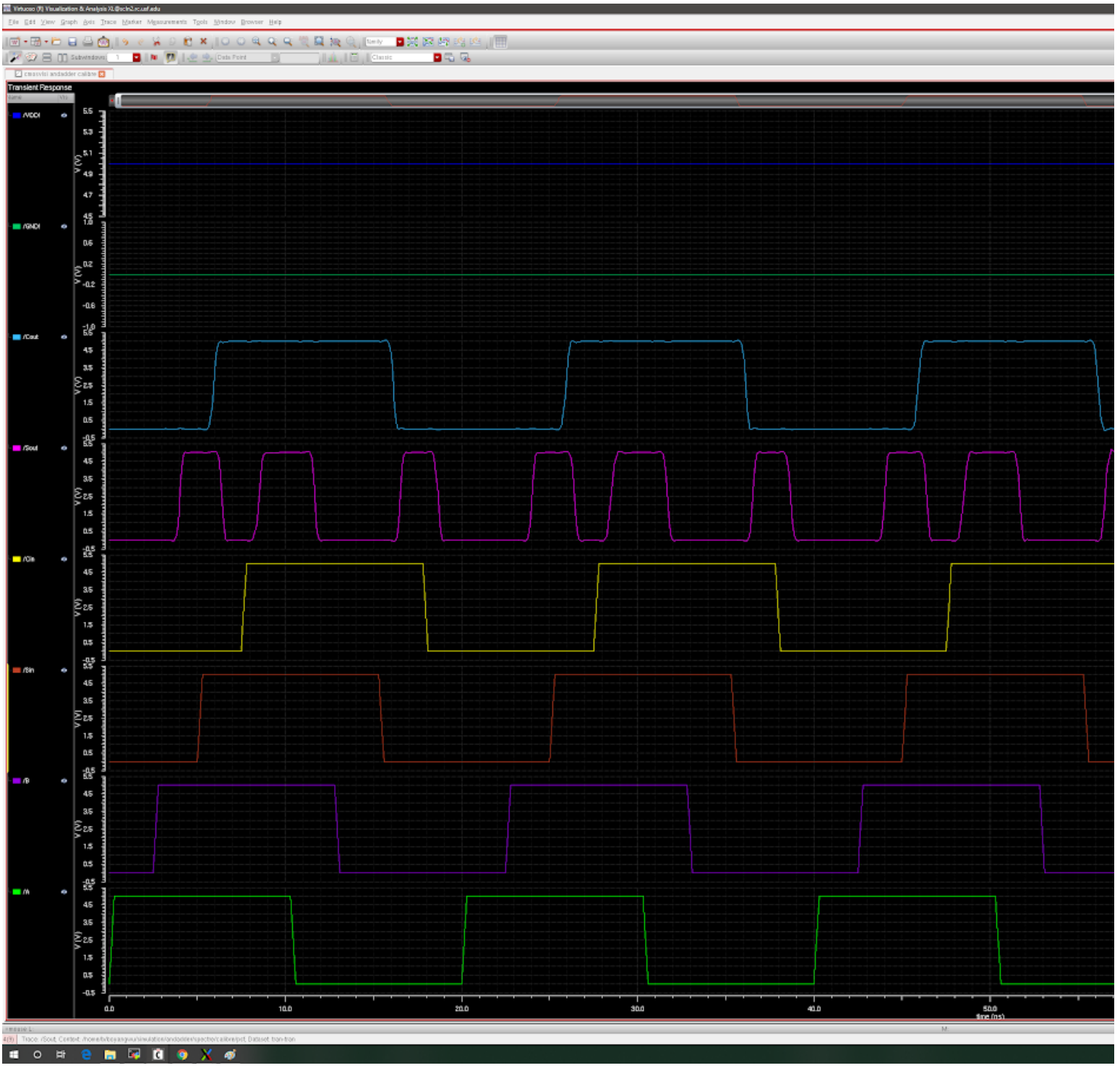
(3)**(25pts)** Simulation Results (without padframe):

(a) (5 pts total) Individual cells:

I. Full Adder:



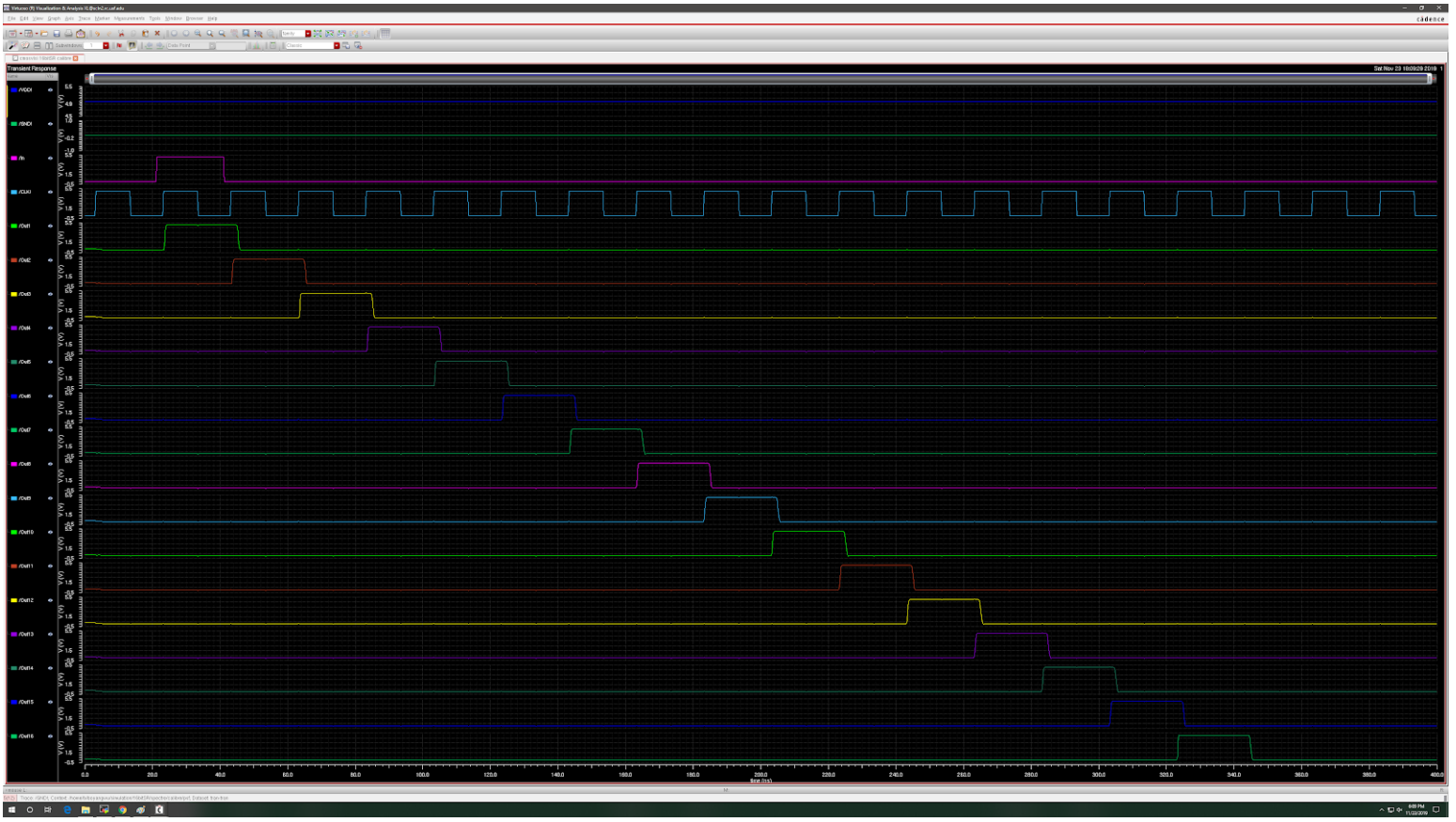
II. Full Adder with AND gate:



III. Registers (Test Mode):

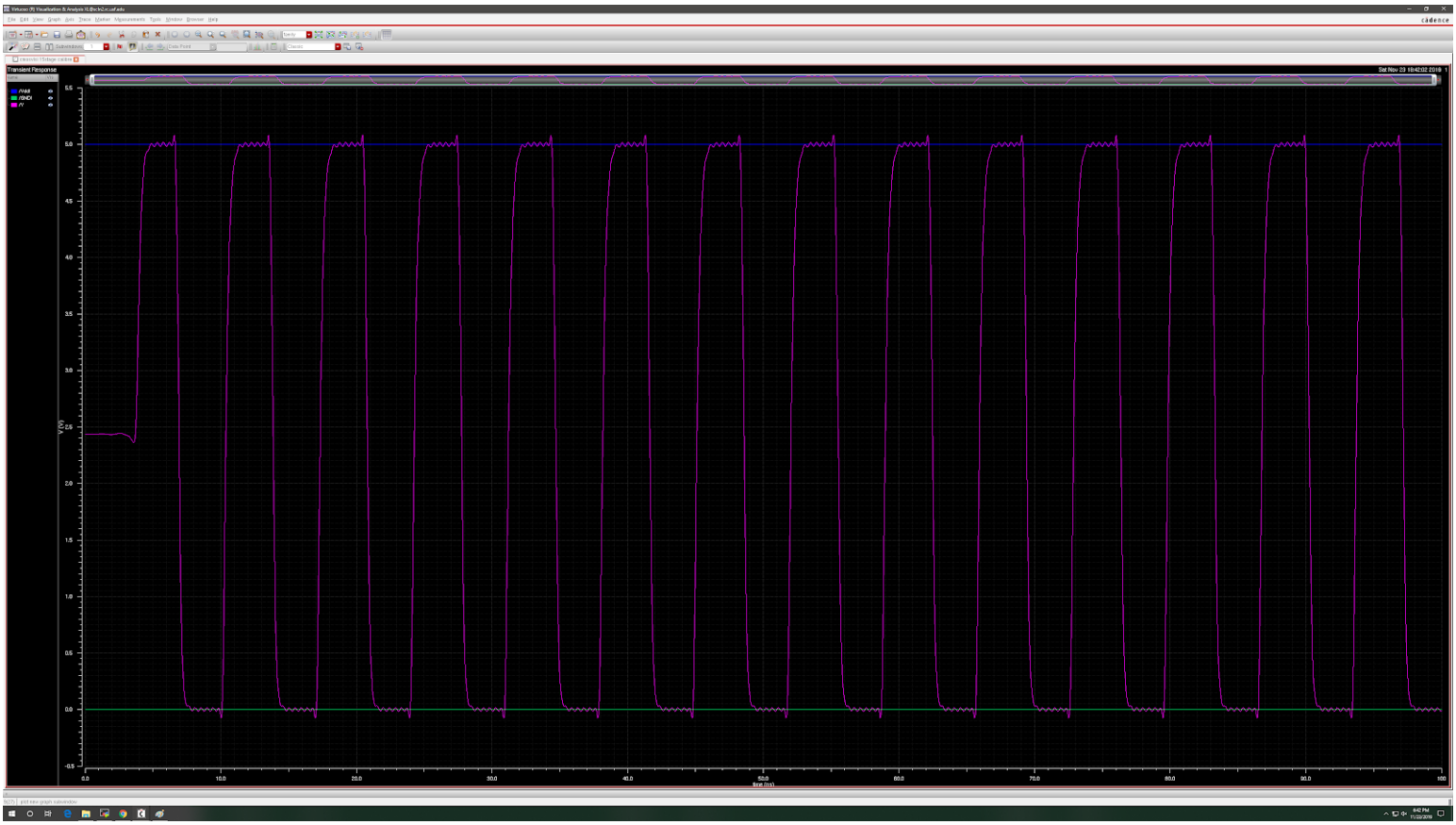
8-bit shift register

16-bit shift register



IV. Ring Oscillator (If used):

Ended up not being used due to lack of time

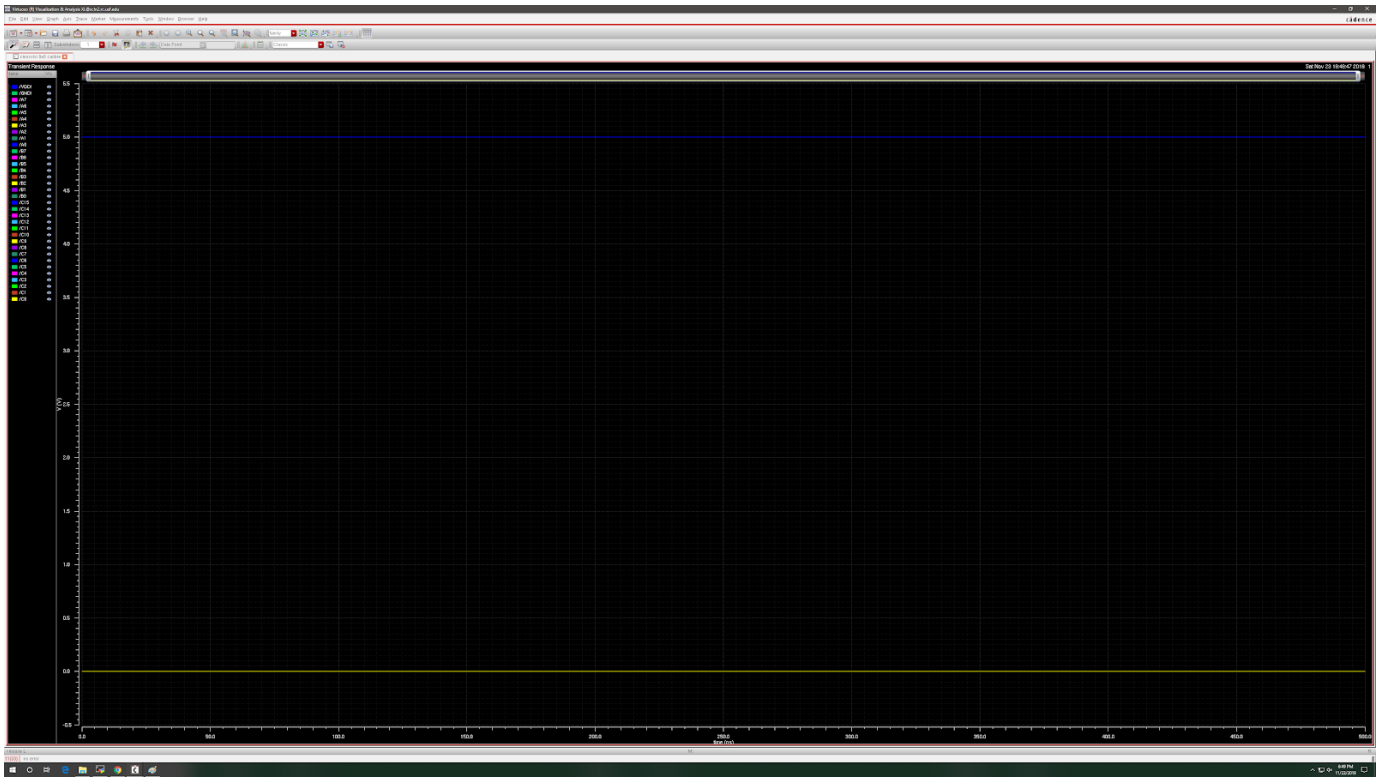


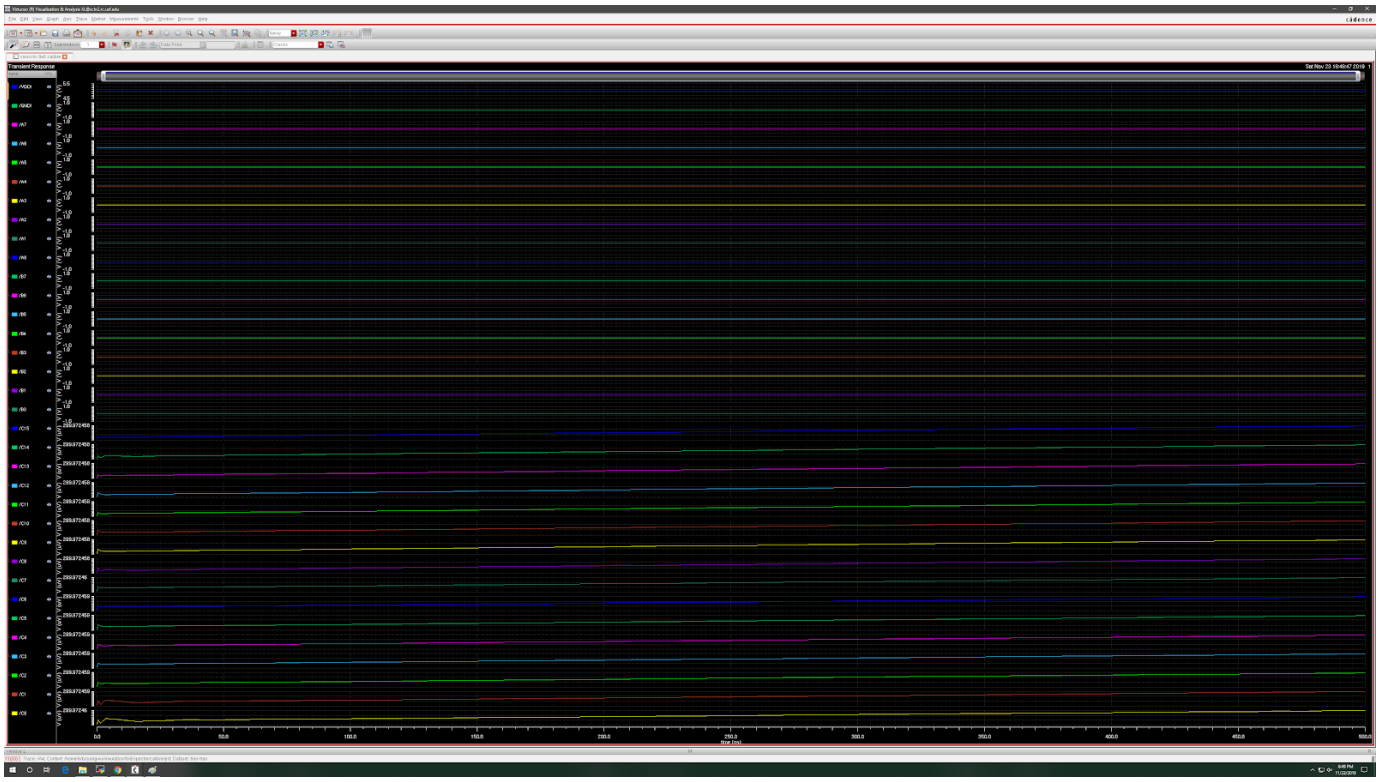
7ns delay

(b) (20 pts) The final multiplier:

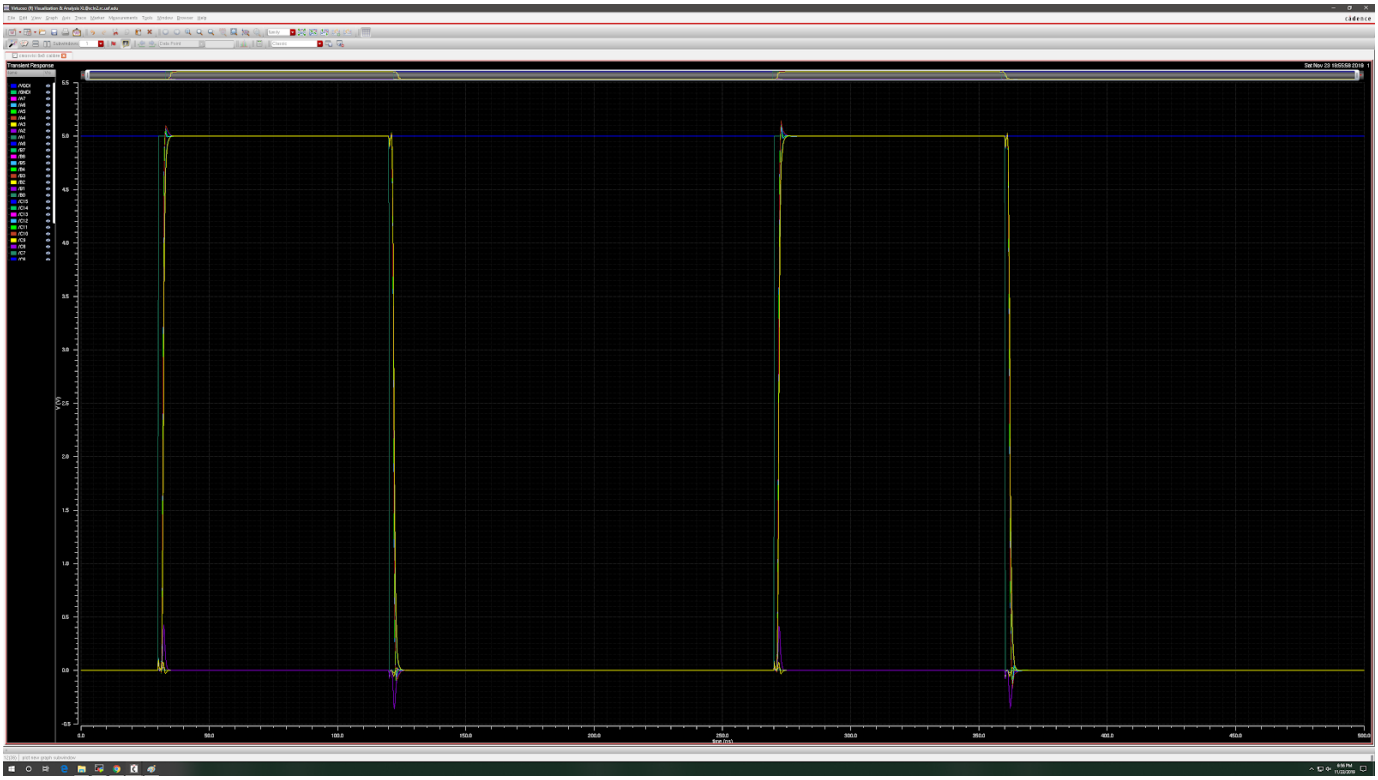
- **This section has not been changed for the final submission. The full 8x8 along with shift registers do work together though as you will see in the next section (with padframe)**
- Parallel in/out are not currently working so look at the waveforms between 40ns to 115ns for correct input.
- There is incorrect input of around 10ns or so (critical path delay) right when inputs switch from 0 to 1 (around 30ns) which will be accounted for in the final design.
- The separated waveforms will also look bad since the Y-axis scales are usually in the mV or less, so we will show waveforms of the combined (to show strong 0's (0V) and 1's (5V)) as well as the separated (to see each individual input/output line). Apologies in advance since it may be harder to read. Zoom in since the original screenshots should be fairly detailed but images imported to document size would be smaller.
- There are also some weird things happening to zeroed input values but they are all in the aV range, so everything is still strong 0's and 1's.

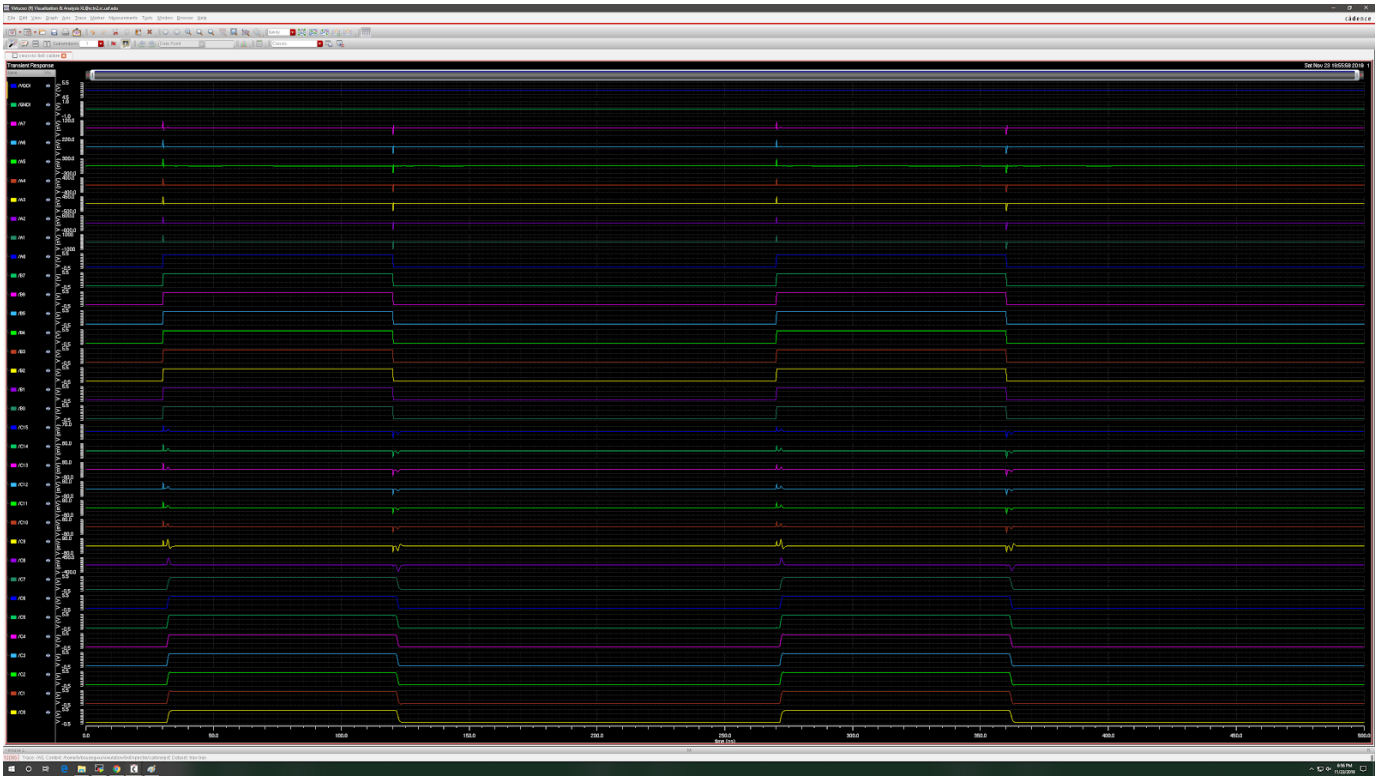
00000000x00000000 = 0000000000000000 (0 x 0 = 0)



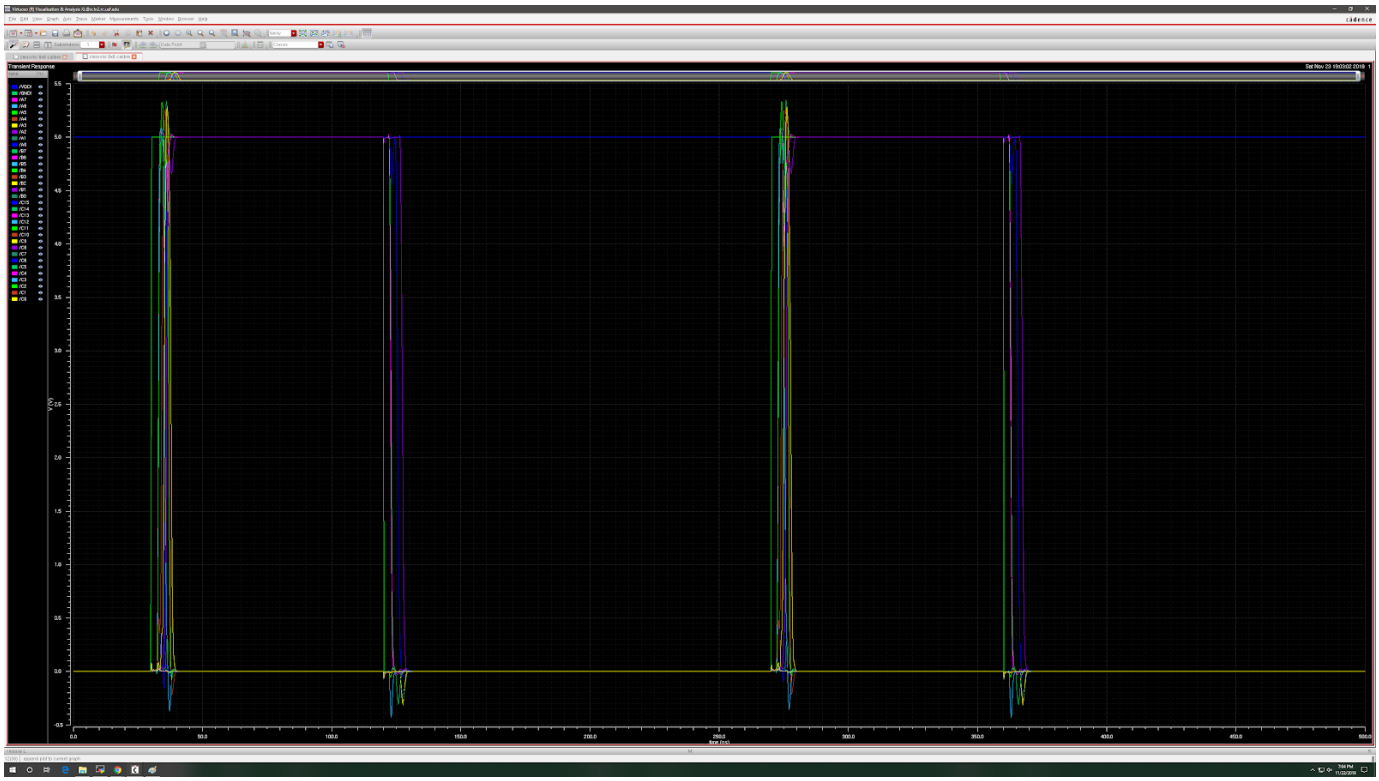


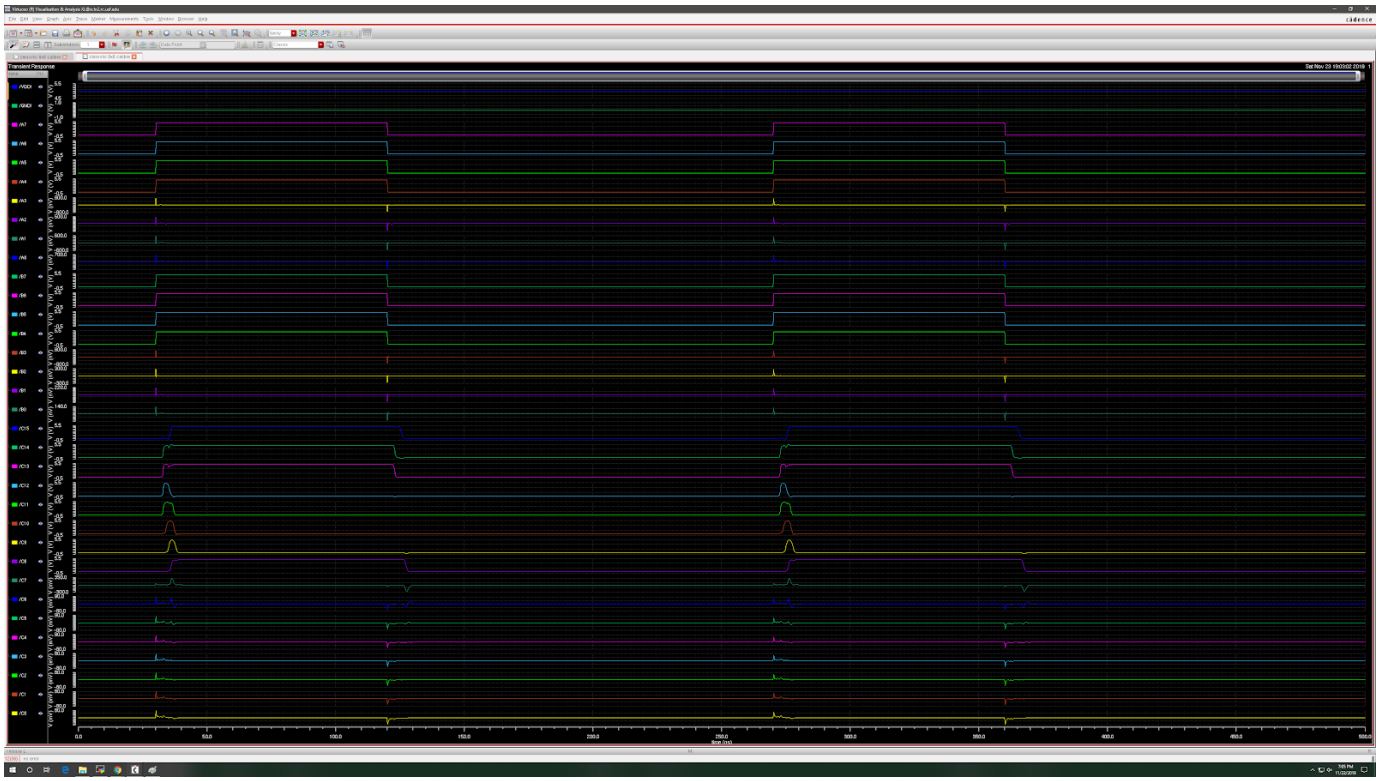
00000001 x 11111111 = 0000000111111111 (1 x 255 = 255)



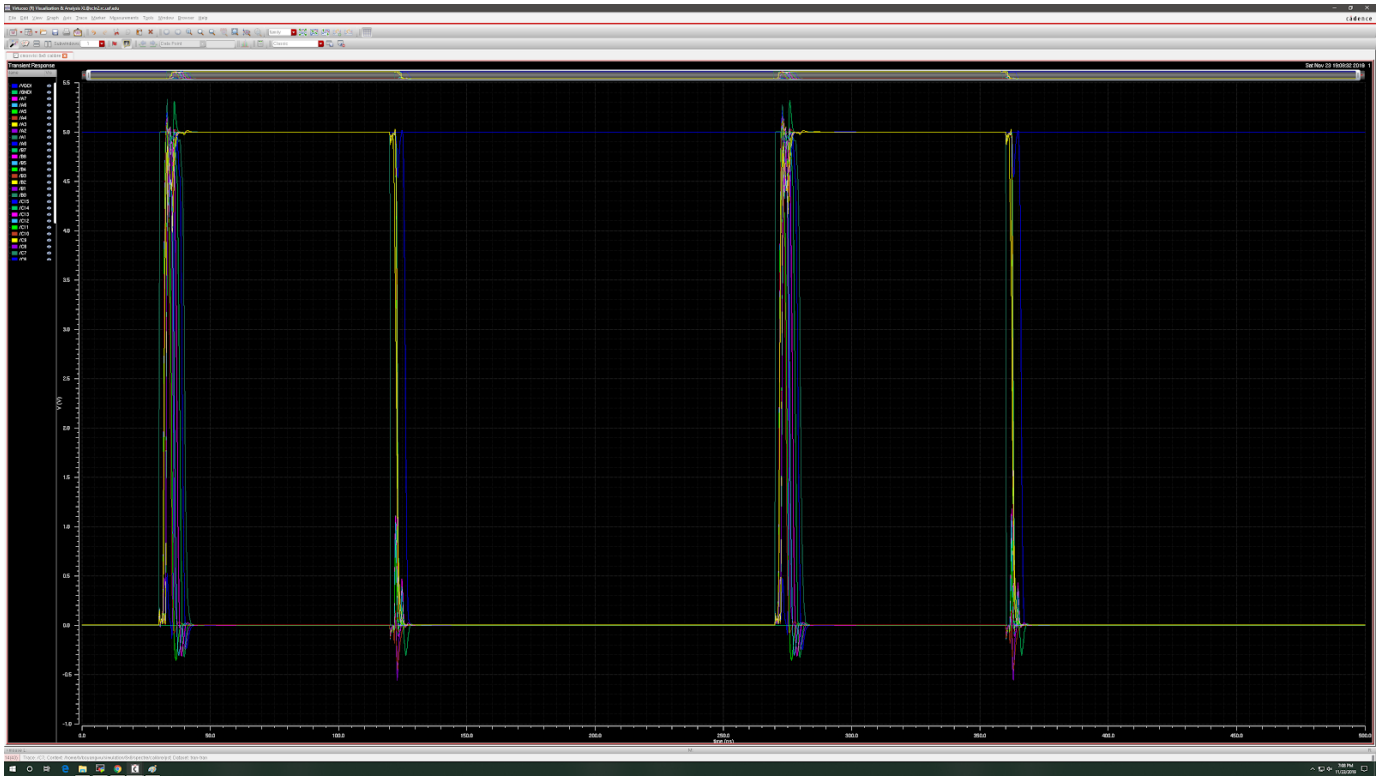


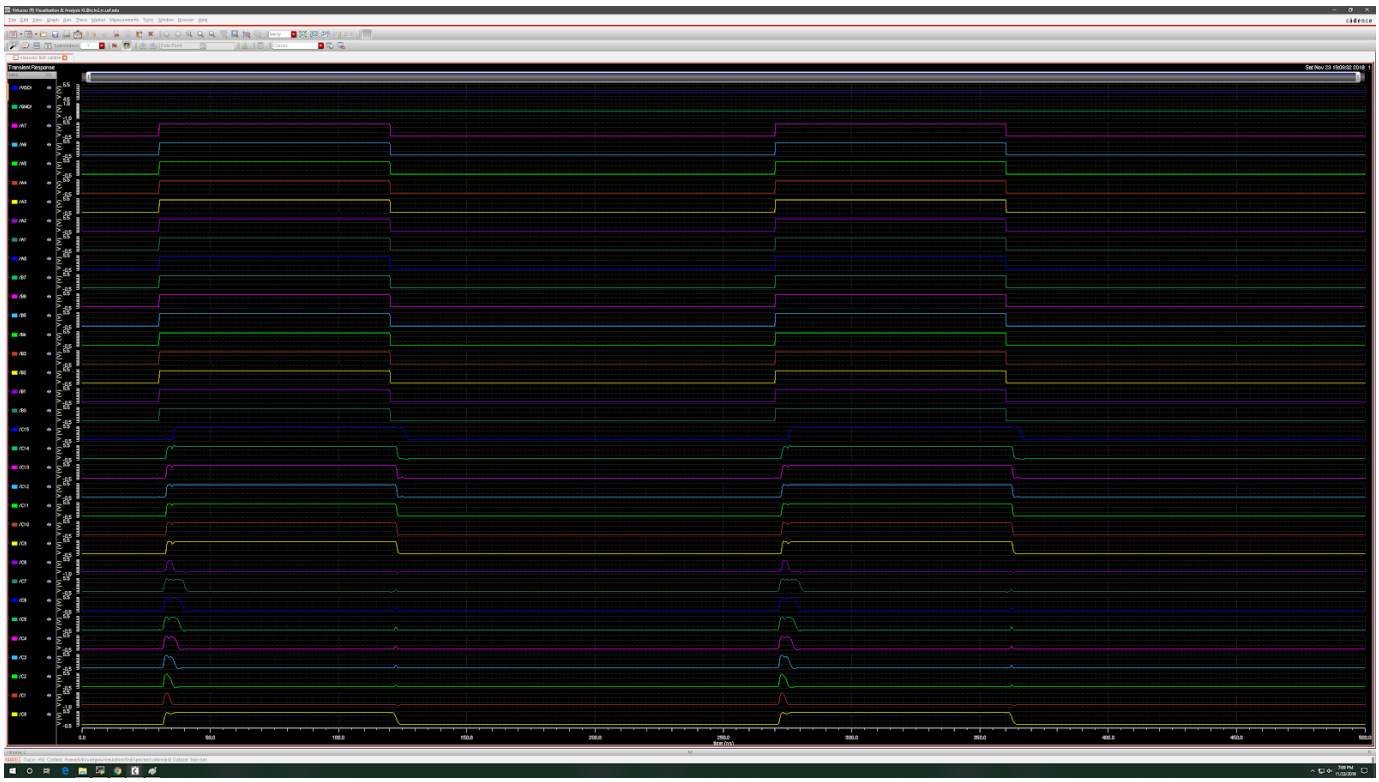
11110000 x 11110000 = 1110000100000000 (240 x 240 = 57600)





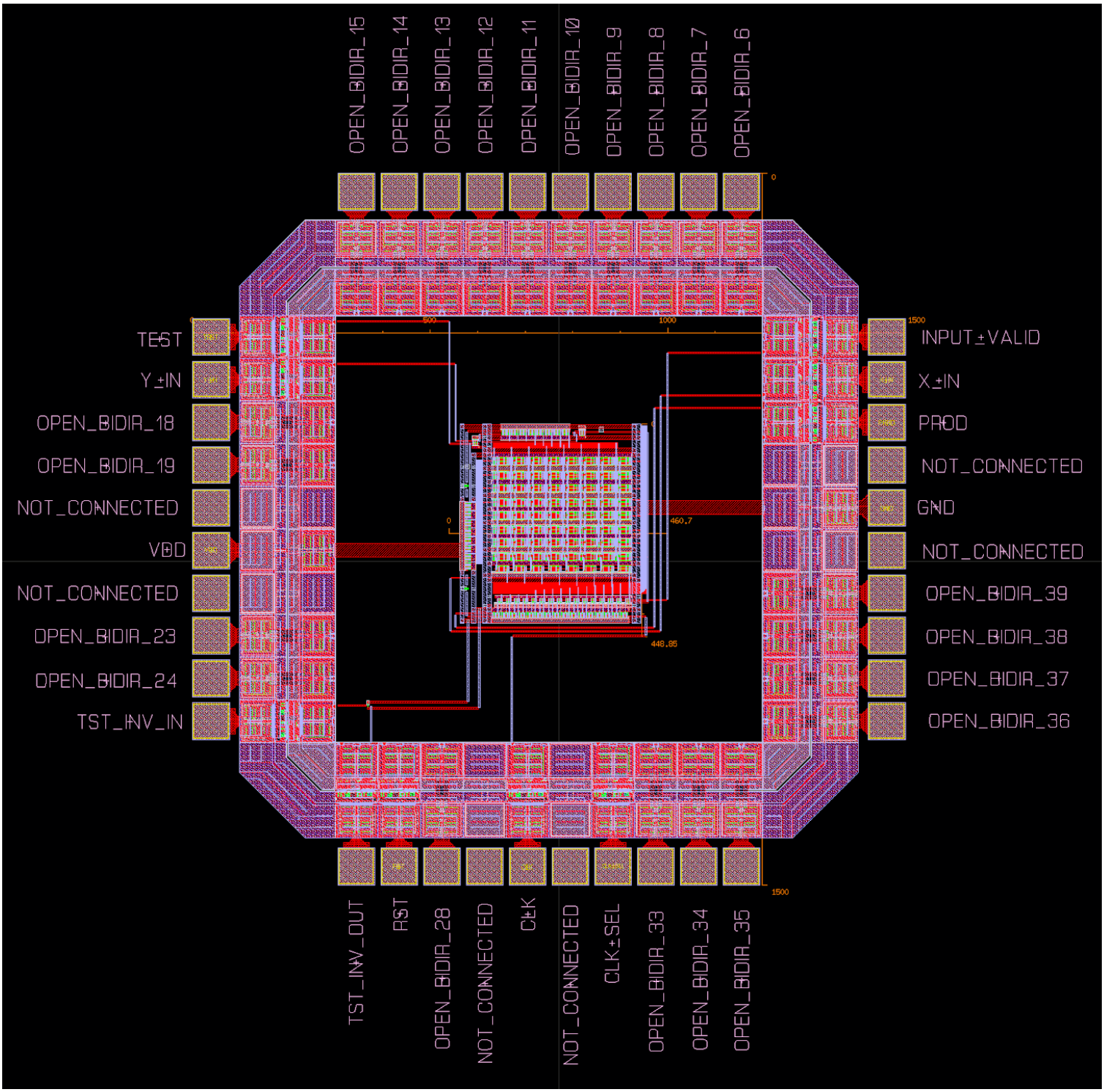
11111111 x 11111111 = 1111111000000001 (255 * 255 = 65025)



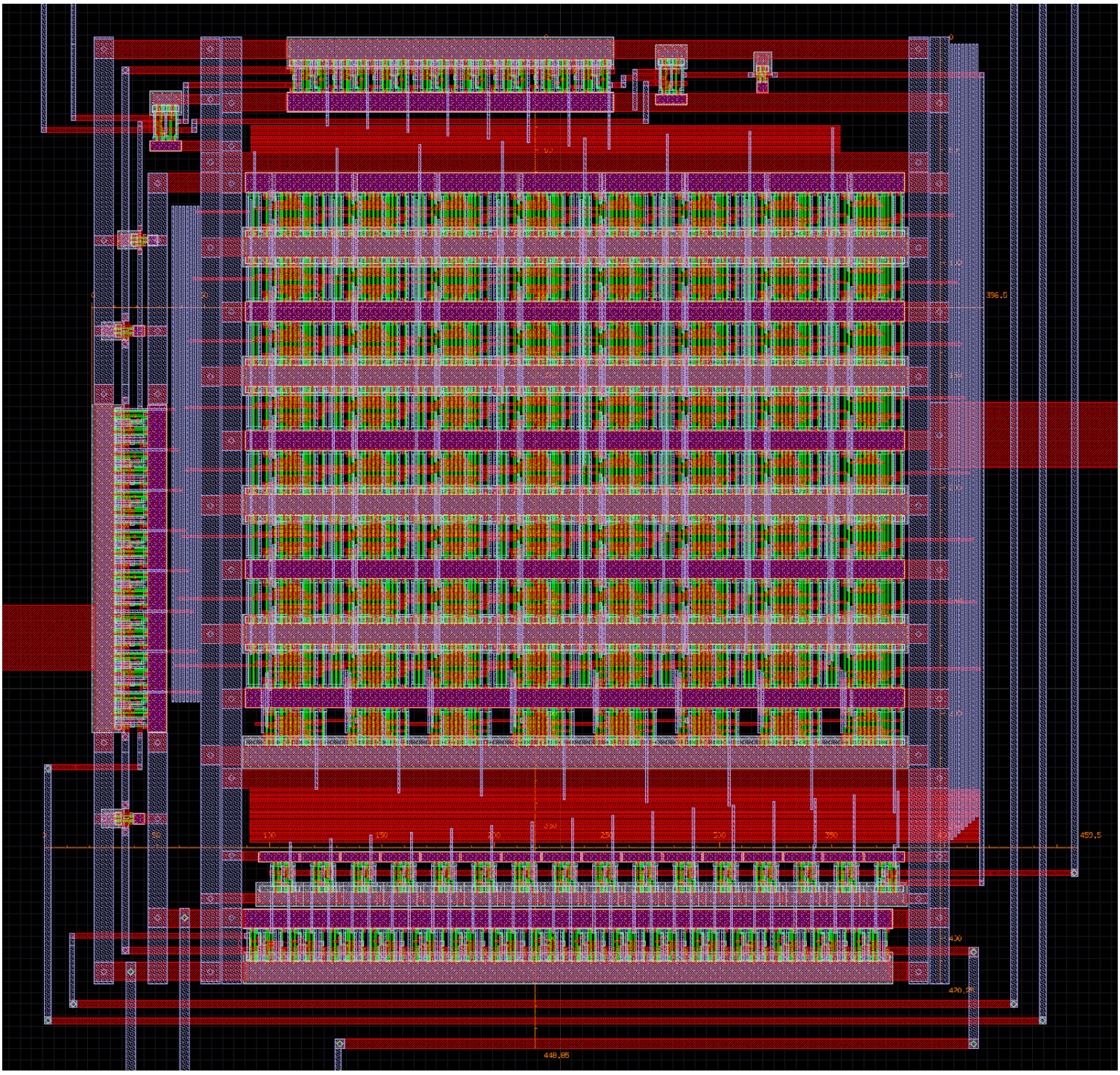


(4)**(10 pts.)** Layout of the final design (with padframe):

Full Padframe



Zoomed In



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

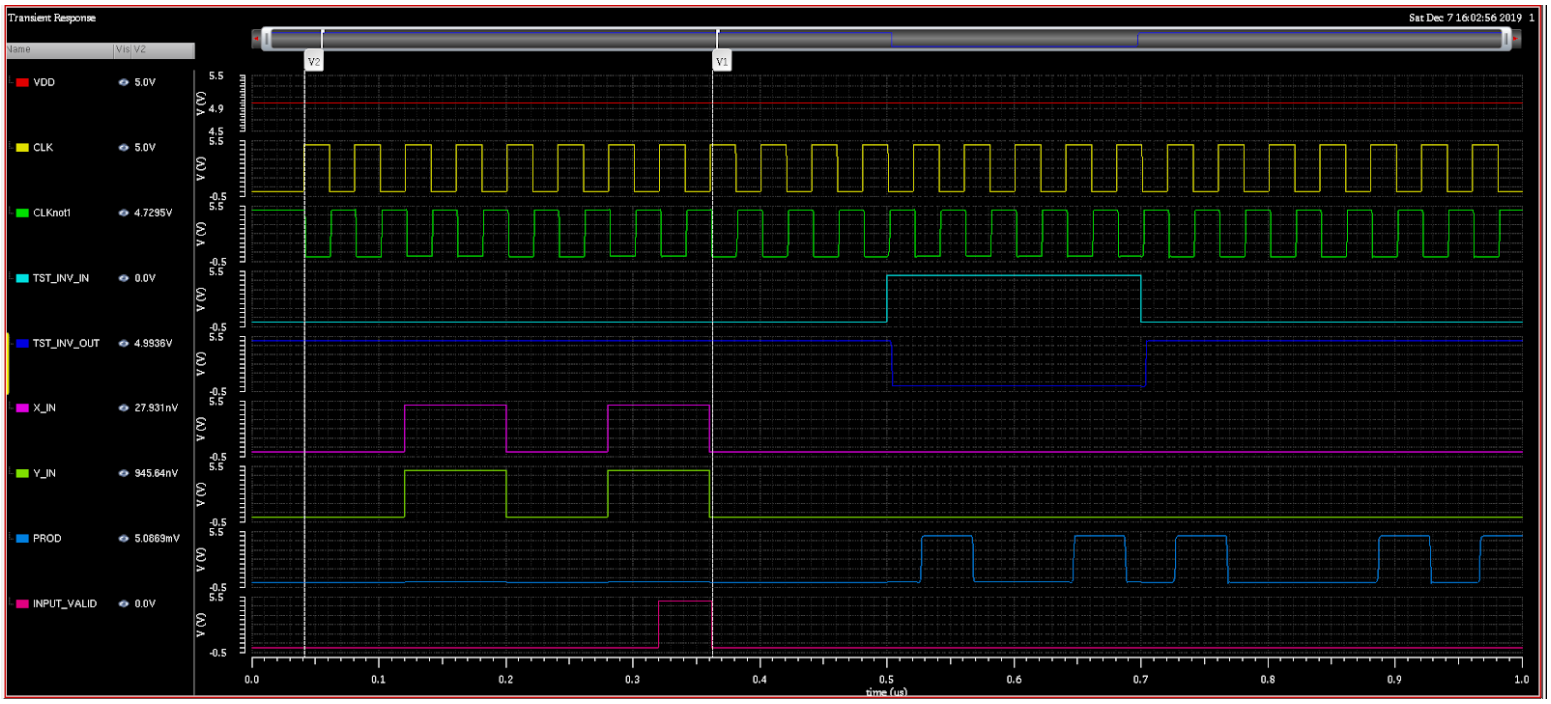
- In the padframe everything gets delayed by 40ns at the start since there were some weird issues going on when certain inputs started off as 1.
- GND is not visible since when using high-performance simulation there seems to be something with Parasitics where any solid 0 input will not show up in the Waveform. VDD is always solid at 5V.
- X_IN and Y_IN start inputting at 40ns and are least significant bit on the left (40ns side) and most significant bit on the right (360ns side).
- INPUT_VALID is enabled for one clock cycle (40ns) from 322ns to 362ns and has to stop exactly at 362ns or the output will not be correct. This input is connected to the 16 Muxes connected to the 16-bit Shift Register (SR), so when INPUT_VALID is High, the 16-bit SR takes in parallel input. When it is Low, the SR is in serial mode to to to PROD (Final serial Output)
- TST_INV_IN and TST_INV_OUT are also on all waveforms to show the test inverter works. There is just a 200ns pulse from around 500ns to 700ns.
- CLK is the input from the padframe. CLKnot1 was a pin to monitor the inverted clock but should have been removed since it was mainly for debugging.

01010101 x 01010101 = 0001110000111001 (85 x 85 = 7225)

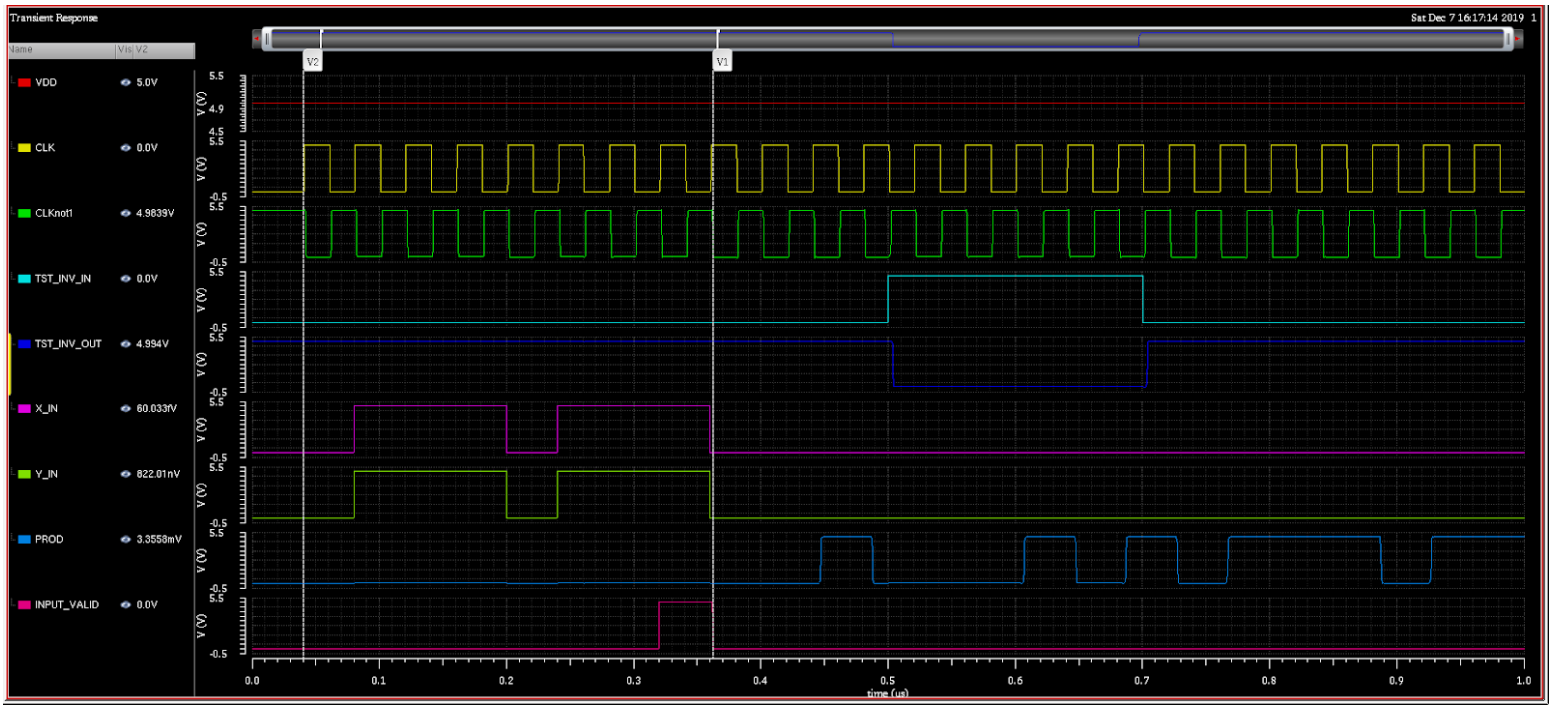
10001000 x 10001000 = 0100100001000000 (136 x 136 = 18496)

10101010 x 10101010 = 0111000011100100 (170 x 170 = 28900)

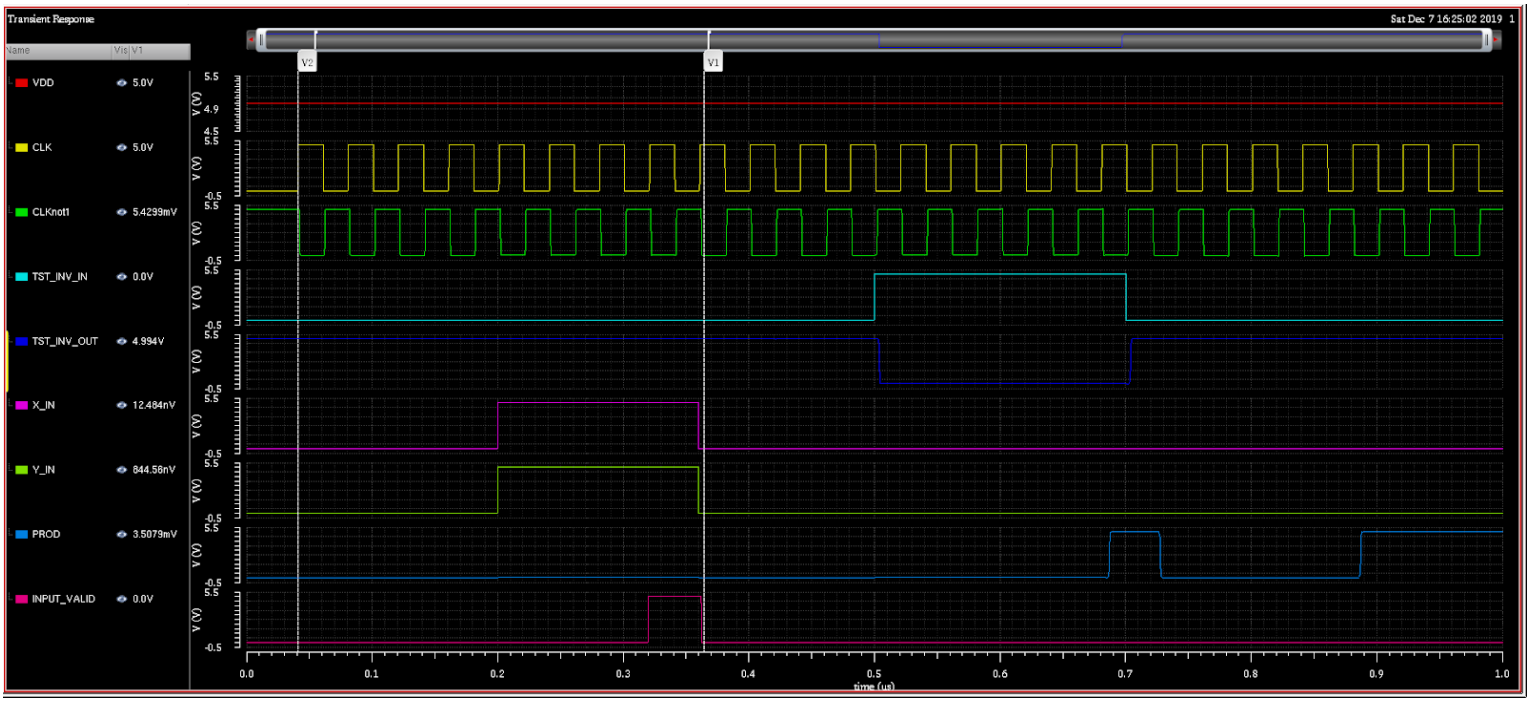
11001100 x 11001100 = 1010001010010000 (204 * 204 = 41616)



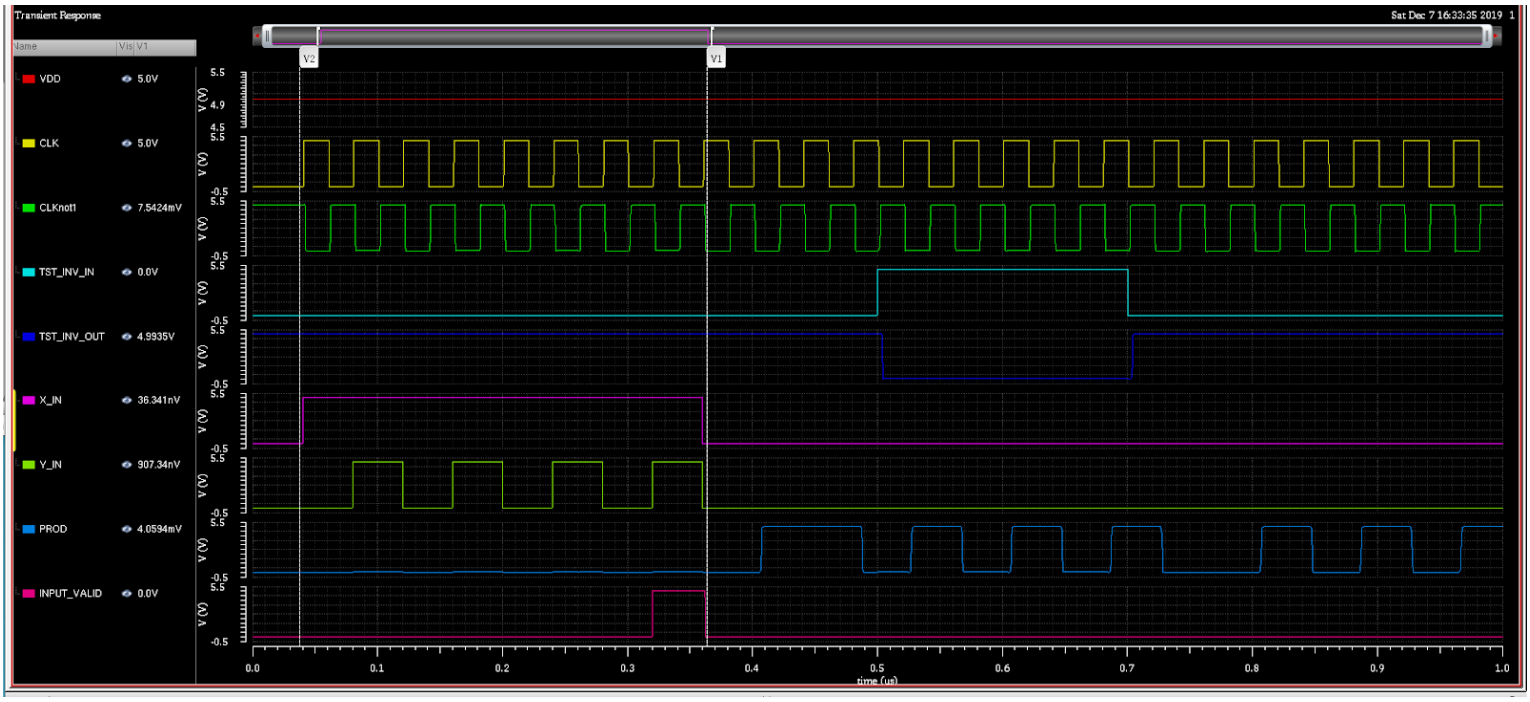
11101110 x 11101110 = 1101110101000100 (238 x 238 = 56644)



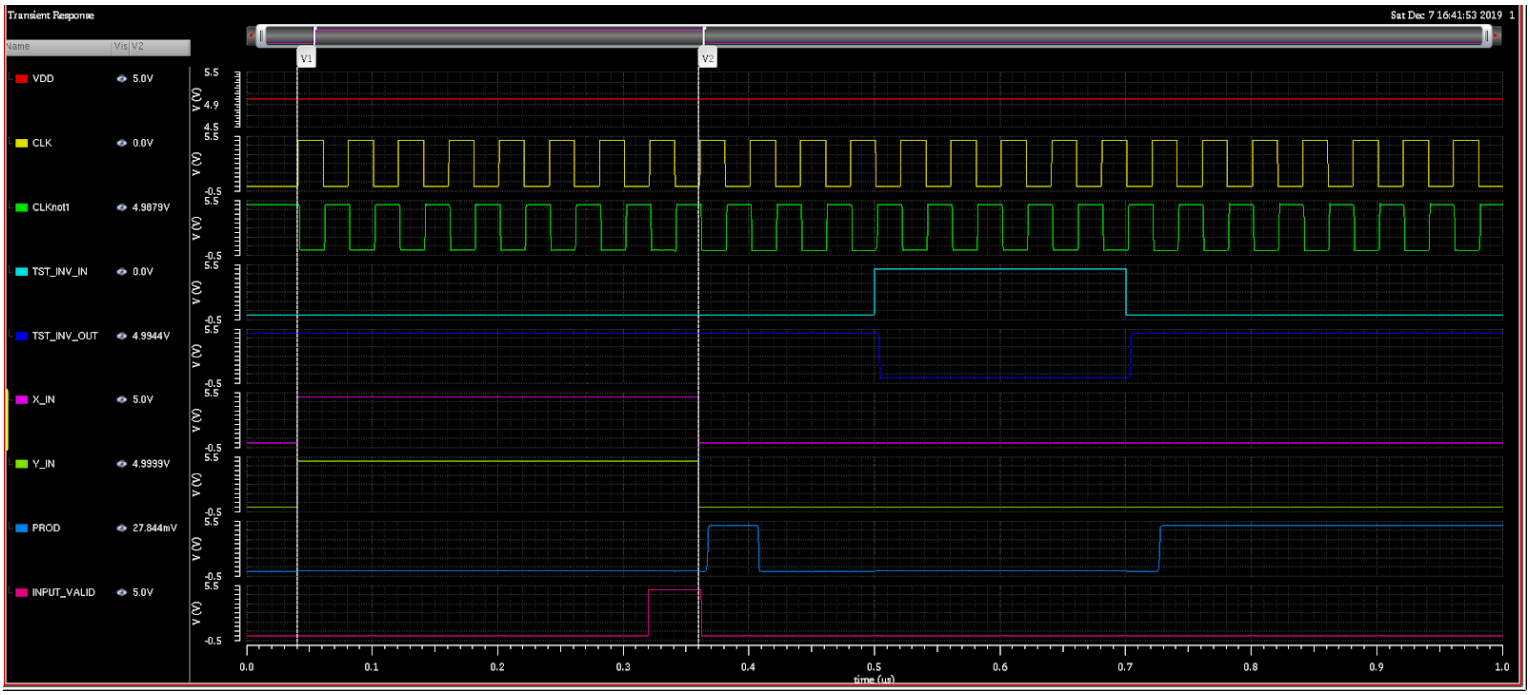
11110000 x 11110000 = 1110000100000000 (240 x 240 = 57600)



11111111 x 10101010 = 1010100101010110 (255 x 170 = 43350)

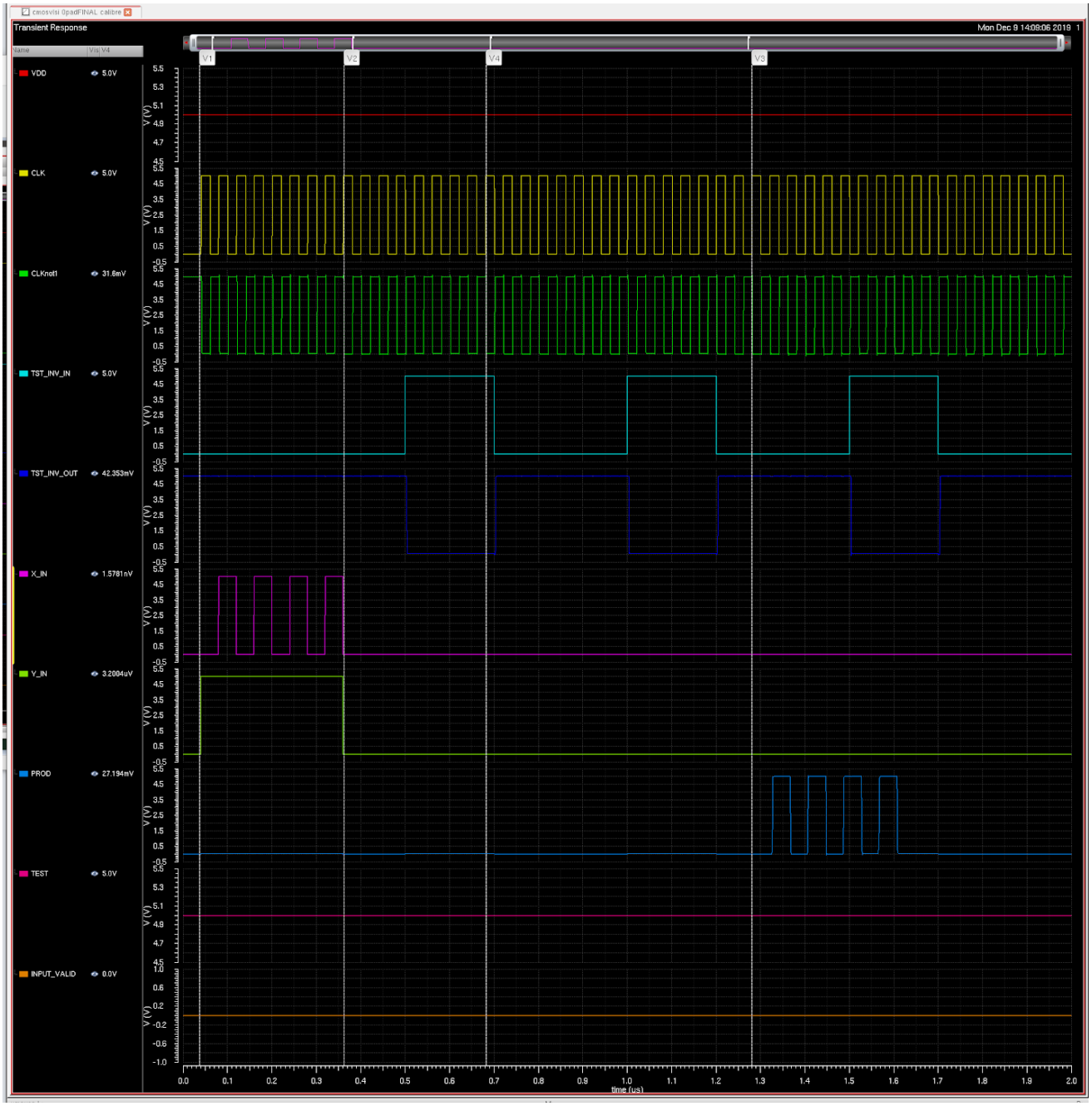


11111111 x 11111111 = 1111111000000001 (255 x 255 = 65025)

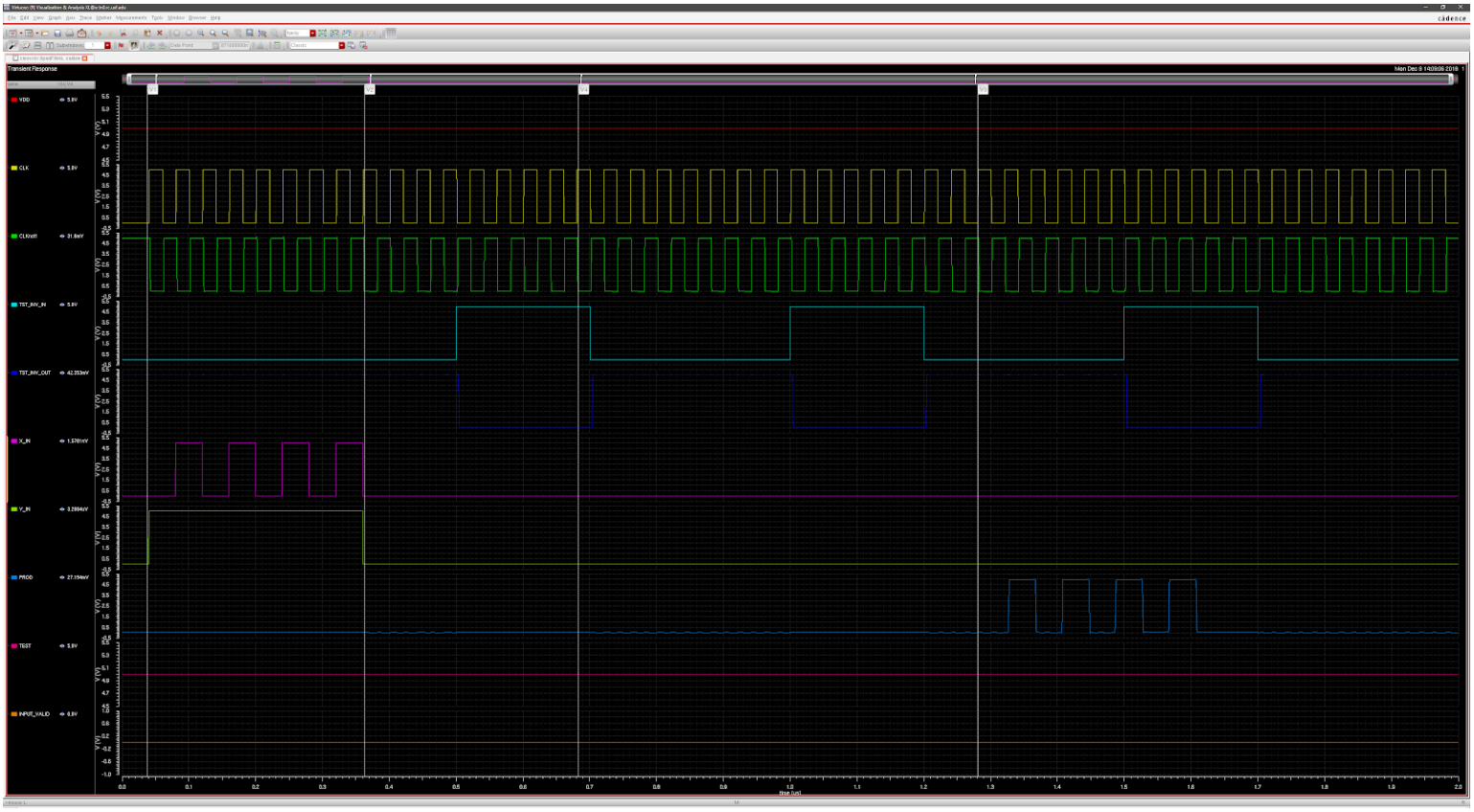


Test Mode 1

- X_IN is 10101010 from 360ns to 40ns. Y_IN is 11111111 but since it is test mode it does not do anything. TEST is High the entire time to enable test mode. INPUT_VALID is Low.
- V1 to V2 are the 8 clock cycles needed for the 8-bit input to go into the first 8-bit shift Register (from X_IN).
- V2 to V4 (3rd white line) are where the 2nd 8-bit Shift Register should be getting serial input, but of course no pins are there to show it.
- V4 to V3 (4th line) would be where the 16-bit Shift Register gets serial input. Exactly on the 16th clock cycles at V3 is when PROD (Serial out) is getting values and the waves show that.

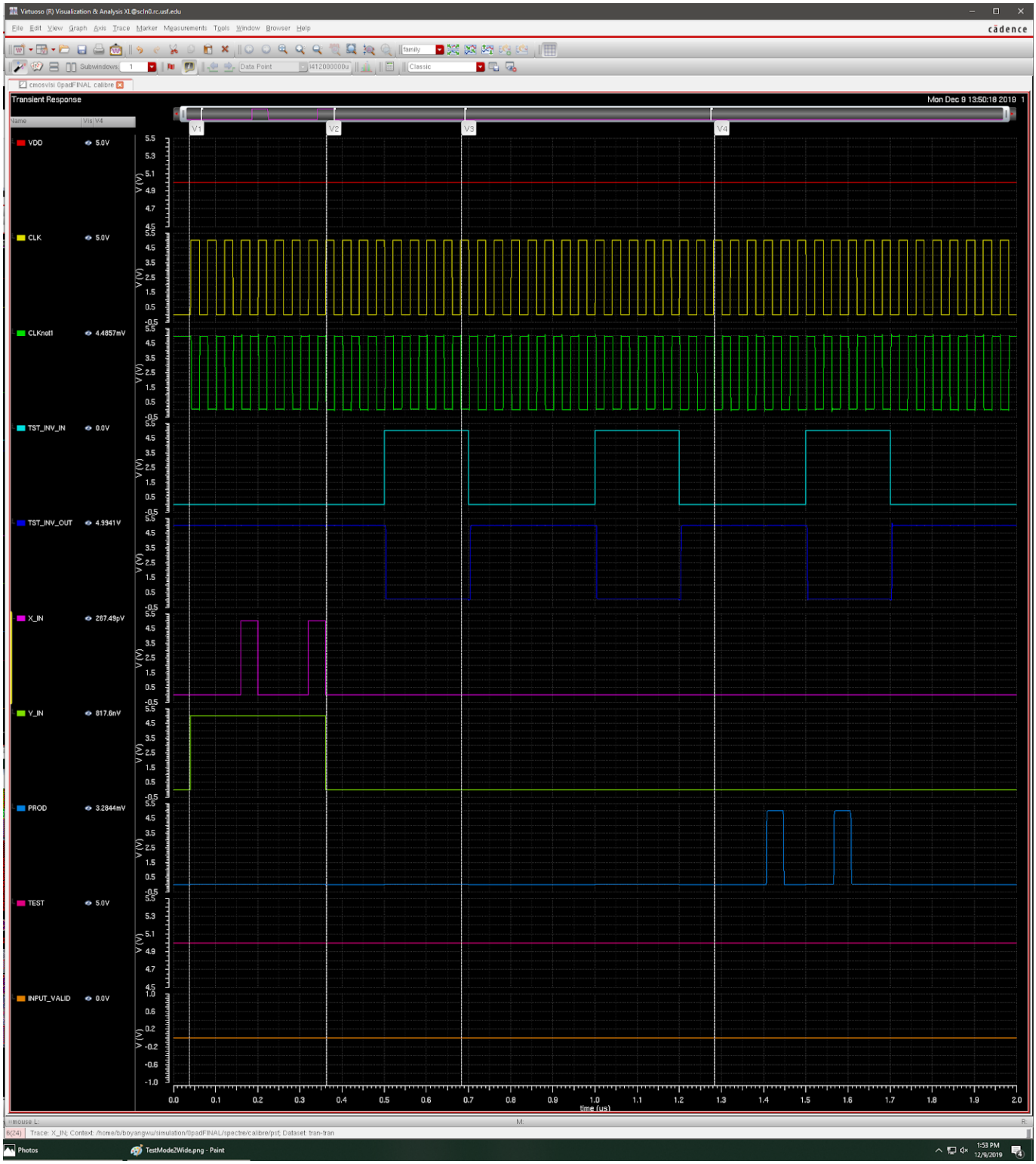


Still Test Mode 1 but with a wider graph



Test Mode 2

- Same thing as above but V3 and V4 are swapped (they are in the correct order now)
- X_IN is now 10001000 and Y_IN is still the same at 11111111 but Y_IN does nothing.



Still Test Mode 2 but with a wider graph