

Digital Microelectronics

POLITECNICO DI TORINO

DIPARTIMENTO DI ELETTRONICA E TELECOMUNICAZIONI

Final Project:
**Standard Cell layout of an inverter and a
half-adder**

Group 5:

Marco Andorno (247222)

Michele Caon (253027)

Matteo Perotti (251453)

Date: June 16, 2018

1 Introduction

The goal of this final project is to design and characterize two logic gates of a standard cell library. These are in particular an inverter whose transistors have width four times the minimum, and therefore a 4 times the minimum drive strength (INV_X4 in the following), and a half adder with minimum drive strength (HA_X1 in the following).

The design starts with the schematic drawing, using an available .png image as a reference, which has to be simulated as an initial condition. Then, the layout of the cell must be drawn and the parasitic elements must be extracted from it. Finally, those parasitics must be used for the final characterization in order to fill in the Liberty file of the standard cell library.

The files for these two gates are inside the following paths:

/home/md18.5/project/INV_X4/

and

/home/md18.5/project/HAX1/

Note that all the material, like scripts and the current report files, can be found in the following GitHub repository:

<https://github.com/mksoc/standard-cell-design>

2 Inverter

2.1 Schematic

Starting from the schematic of the available .png (figure 1), we copied it using Virtuoso Schematic Editor, obtaining the result shown in figure 2. Note that the four PMOS and NMOS transistors in parallel will be just one large transistor in the layout, with the source/drain diffusions fingered.

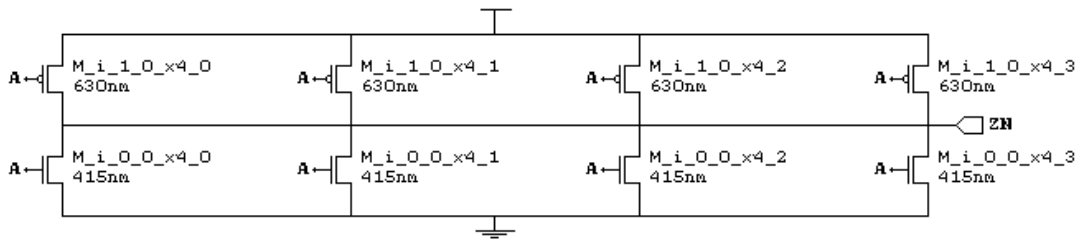


Figure 1: Reference schematic of the inverter

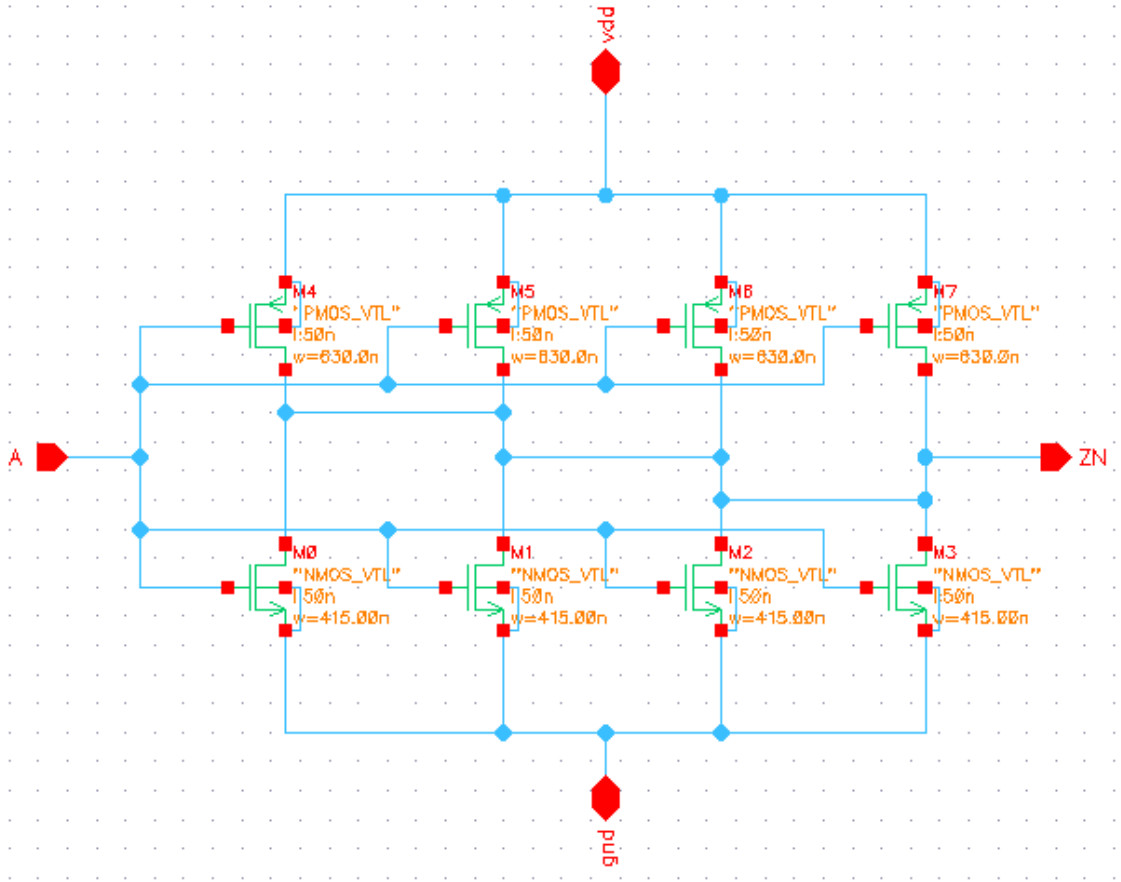


Figure 2: Final schematic of INV_X4

We then carried out some preliminary simulations using a test bench, to confirm that the circuit works correctly. We measured rise and fall times, as well as propagation delays, that will be compared with the actual characterization later on. We also measured the transfer characteristic, shown in figure 3.

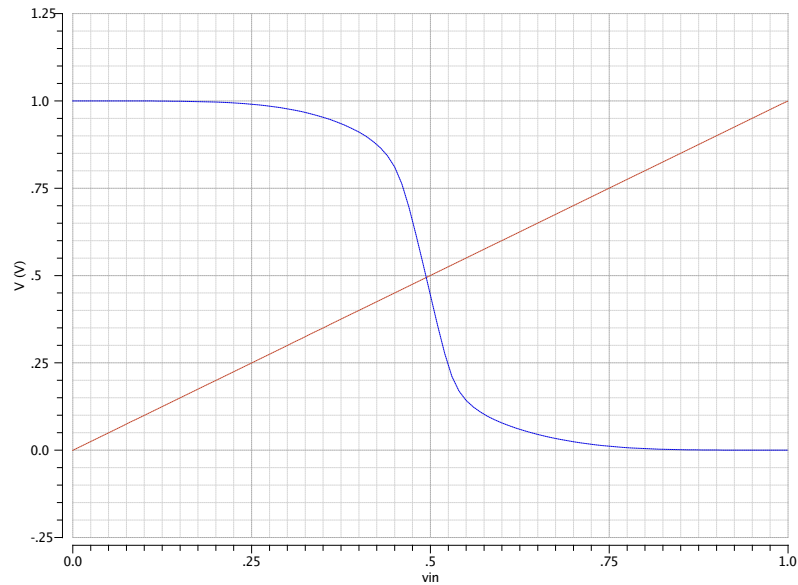


Figure 3: Inverter transfer characteristic

2.2 Layout

The final layout of INV_X4 is shown in figure 4. The main trick to note here is how we used transistor fingering extensively, by sharing as much as possible source and drain diffusions. This is generally done in order to reduce the total area of the standard cell, but in this case it was actually mandatory, because a single transistor large 4 times the minimum width would not fit inside the well height, at least for the PMOS pull-up network.

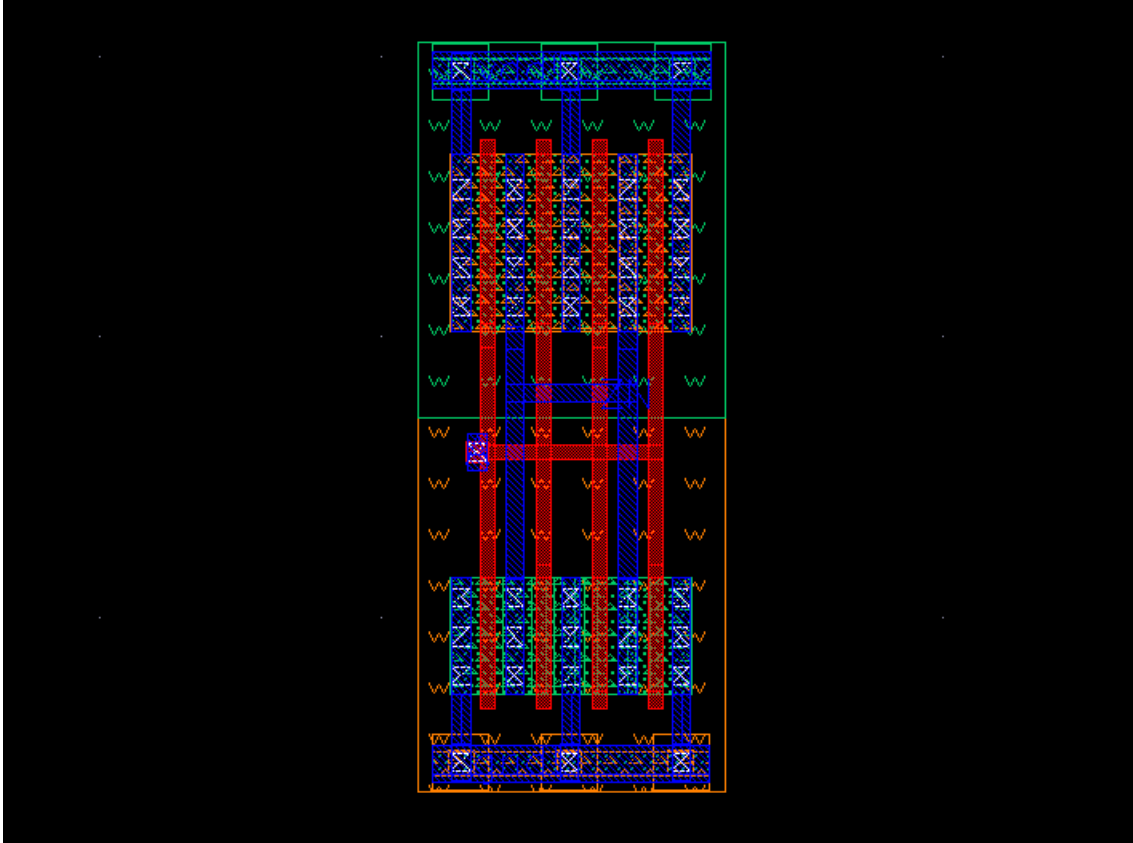


Figure 4: Layout of INV_X4

For PMOS transistors we were forced to finger four different transistor, while regarding the NMOS, probably, two fingered transistors with double width would have fit, but since the cell width would have been the same, we decided to go with four single width transistors anyway to improve symmetry.

Regarding pins, the output ZN did not cause any troubles and was placed on the rightmost metal strip connecting the drains of the PMOS and NMOS. As for the input A, instead, our first idea had been to place its via in the middle of the horizontal polysilicon strip, but then we noted that this way the external connections to that pin would have needed to cross the metal 1 layer, thus requiring metal 2 to be used instead. So eventually we settled with placing the via for the input on the leftmost part of the polysilicon, that does not force any metal 1 crossing and also more accurately reflects the topology of the schematic.

While developing the layout we ran the DRC tool many times to ensure all the design rules were met. After completing it we got the following final statement:

```

=====
=== CALIBRE::DRC-H SUMMARY REPORT
===
Execution Date/Time:      Wed May 23 17:11:38 2018
Calibre Version:         v2011.4_14.13      Tue Nov 15 15:18:31 PST 2011
Rule File Pathname:      /home/md18.5/project/INV_X4/_calibreDRC.rul_
Rule File Title:
Layout System:           GDS
Layout Path(s):          INV_X4.calibre.db
Layout Primary Cell:      INV_X4
Current Directory:        /home/md18.5/project/INV_X4
User Name:               md18.5
Maximum Results/RuleCheck: 1000
Maximum Result Vertices: 4096
DRC Results Database:    INV_X4.drc.results (ASCII)
Layout Depth:            ALL
Text Depth:              PRIMARY
Summary Report File:     INV_X4.drc.summary (REPLACE)
Geometry Flagging:       ACUTE = NO  SKEW = NO  ANGLED = NO  OFFGRID = NO
                          NONSIMPLE POLYGON = NO  NONSIMPLE PATH = NO

Excluded Cells:
CheckText Mapping:       COMMENT TEXT + RULE FILE INFORMATION
Layers:                  MEMORY-BASED
Keep Empty Checks:       YES
-----
--- RUNTIME WARNINGS
---
-----
--- ORIGINAL LAYER STATISTICS
---
LAYER pwell ..... TOTAL Original Geometry Count = 4  (4)
.
.
.
LAYER via9 ..... TOTAL Original Geometry Count = 0  (0)
-----
--- RULECHECK RESULTS STATISTICS
---
RULECHECK Well1.1 ..... TOTAL Result Count = 0 (0)
.
.
.
RULECHECK Antenna.metal10 ... TOTAL Result Count = 0 (0)
-----
--- RULECHECK RESULTS STATISTICS (BY CELL)
---
-----
--- SUMMARY
---
TOTAL CPU Time:          0
TOTAL REAL Time:         0
TOTAL Original Layer Geometries: 171 (171)
TOTAL DRC RuleChecks Executed: 167
TOTAL DRC Results Generated: 0 (0)

```

Moreover, we compared layout and schematic netlists by using LVS, that succeeded on the first try with its reassuring smiling smiley-face:

```
#####
##                                ##
##          C A L I B R E      S Y S T E M      ##
##                                ##
##          L V S    R E P O R T      ##
##                                ##
#####
REPORT FILE NAME:      INV_X4.lvs.report
LAYOUT NAME:          /home/md18.5/project/INV_X4/INV_X4.sp ('INV_X4')
SOURCE NAME:          /home/md18.5/project/INV_X4/INV_X4.src.net ('INV_X4')
RULE FILE:            /home/md18.5/project/INV_X4/_calibreLVS.rul_
RULE FILE TITLE:      LVS Rule File for FreePDK45
CREATION TIME:        Tue May 29 15:46:49 2018
CURRENT DIRECTORY:    /home/md18.5/project/INV_X4
USER NAME:            md18.5
CALIBRE VERSION:      v2011.4_14.13    Tue Nov 15 15:18:31 PST 2011

                        OVERALL COMPARISON RESULTS
                        #####
                        #          #          #          -    -
                        #          #          #          *    *
                        #  #          #  CORRECT  #          |
                        #  #          #          #          \___/
                        #          #####

*****
                        CELL SUMMARY
*****
Result      Layout      Source
-----
CORRECT      INV_X4      INV_X4
*****

                        LVS PARAMETERS
*****
o LVS Setup:
.
.
.

                        CELL COMPARISON RESULTS ( TOP LEVEL )
                        #####
                        #          #          #          -    -
                        #          #          #          *    *
                        #  #          #  CORRECT  #          |
                        #  #          #          #          \___/
                        #          #####

LAYOUT CELL NAME:      INV_X4
SOURCE CELL NAME:      INV_X4
-----
INITIAL NUMBERS OF OBJECTS
-----
Layout      Source      Component Type
-----
Ports:      4          4
Nets:      4          4
Instances:  4          4      MN (4 pins)
           4          4      MP (4 pins)
-----
Total Inst:  8          8
NUMBERS OF OBJECTS AFTER TRANSFORMATION
-----
Layout      Source      Component Type
-----
Ports:      4          4
Nets:      4          4
Instances:  1          1      _invv (4 pins)
```

```

-----
Total Inst:      1      1
*****
INFORMATION AND WARNINGS
*****
      Matched      Matched      Unmatched      Unmatched      Component
      Layout      Source      Layout      Source      Type
-----
Ports:           4           4           0           0
Nets:            4           4           0           0
Instances:       1           1           0           0      _invv
-----
Total Inst:      1           1           0           0
o Statistics:
  8 layout mos transistors were reduced to 2.
  6 mos transistors were deleted by parallel reduction.
  8 source mos transistors were reduced to 2.
  6 mos transistors were deleted by parallel reduction.
o Layout Names That Are Missing In The Source:
  Ports:         GND! VDD!
  Nets:          GND! VDD!
o Initial Correspondence Points:
  Ports:         A ZN
*****
SUMMARY
*****
Total CPU Time:      0 sec
Total Elapsed Time:  0 sec

```

Just a note on the warning that says that nets GND! and VDD! are present in the layout but missing in the source (schematic). We tried to follow the steps of the laboratory sessions where the supply nets had the exclamation mark at the end in the layout view but not in the schematic one, and in fact we had no problems with this method for the INV_X4. However, we had a lot of issues for HA_X1 when running LVS as it complained about missing power nets. We finally came to the conclusion that schematic and layout must have the exact same names for supply nets and that the exclamation mark can be avoided if the schematic does not exploit global nets to make the drawing cleaner and clearer. Even then, to this day it remains unclear why the inverter cell worked just as well with different names for power nets (disregarding this warning), while the half adder with the same setup caused so much trouble. We can only accept this and give in to the mighty secrets of Virtuoso.

2.3 Characterization

The first step toward the complete characterization of our logic gate was the extraction of the parasitic elements from the layout. To do this, we ran the PEX tool from the layout editor and got the following report as well as a complete new netlist that takes into account those parasitics in the schematic.

```

#####
##                                     ##
##                               Calibre xRC                               ##
##                                     ##
##                               Export Lumped Parameters                 ##
##                                     ##
#####
LAYOUT NAME:      INV_X4
RULE FILE NAME:   rules
CREATION TIME:    Thu May 31 18:19:25 2018
UNITS:            Resistance = ohm

```

CELL NAME: INV_X4					
Netid	R(UpperBound)	Cvalue	%Coupled	RC(UpperBound)	Netname
<hr/>					

8


```

4.8594e-18 netid: MM1:g
1.08097e-18 netid: MM5:g
6.73195e-18 netid: c_13:p
9.48261e-18 netid: MM0:g
1.06043e-17 netid: MM4:g
4.14482e-18 netid: MM5:g
4.99255e-18 netid: MM7:g
2.54537e-17 netid: MM3:s
3.92759e-18 netid: c_70:n
2.3559e-17 netid: MM2:s
6.89683e-19 netid: MM2:s
2.3559e-17 netid: MM2:s
2.07068e-19 netid: c_76:n
3.92759e-18 netid: c_76:n
2.54537e-17 netid: MM0:s
3.86405e-17 netid: MM7:s
2.77537e-18 netid: c_90:n
2.29436e-19 netid: c_90:n
3.66791e-17 netid: MM6:s
3.35031e-18 netid: MM6:s
3.48897e-17 netid: MM6:s
2.77537e-18 netid: c_94:n
3.86405e-17 netid: MM4:s
3 0.0 2.69349e-16 50.1268 0.0 GND
- Coupled nets
  MM3:g,MM3:g,MM2:g,c_6:p,MM2:g,c_9:p,MM1:g,MM1:g,MM0:g,MM0:g,MM3:d,MM3:d,MM3:d,c_43:n,MM1:d,c_43:n
- Intrinsic Capacitance
  1.34333e-16
- Coupled capacitance
  8.27127e-18 netid: MM3:g
  1.34578e-18 netid: MM3:g
  1.35056e-18 netid: MM2:g
  2.57507e-19 netid: c_6:p
  2.18914e-18 netid: MM2:g
  4.78238e-19 netid: c_9:p
  2.18914e-18 netid: MM1:g
  1.35056e-18 netid: MM1:g
  1.4614e-18 netid: MM0:g
  8.27127e-18 netid: MM0:g
  2.54537e-17 netid: MM3:d
  3.92759e-18 netid: MM3:d
  2.3559e-17 netid: MM3:d
  6.89683e-19 netid: c_43:n
  2.3559e-17 netid: MM1:d
  2.07068e-19 netid: c_43:n
  3.92759e-18 netid: MM1:d
  2.54537e-17 netid: MM1:d
  5.04182e-19 netid: MM7:s
  5.69945e-19 netid: MM4:s
4 0.0 3.55081e-16 54.3925 0.0 VDD
- Coupled nets
  MM7:g,MM7:g,MM6:g,MM6:g,MM5:g,MM5:g,MM4:g,MM4:g,MM7:d,MM7:d,c_43:n,MM7:d,c_43:n,MM5:d,MM5:d,MM4:d,MM4:d,c_43:n
- Intrinsic Capacitance
  1.61944e-16
- Coupled capacitance
  1.21306e-17 netid: MM7:g
  1.15069e-18 netid: MM7:g
  1.11503e-18 netid: MM6:g
  3.18871e-18 netid: MM6:g
  3.16413e-18 netid: MM5:g
  7.07137e-19 netid: MM5:g
  4.92603e-19 netid: MM4:g
  1.21341e-17 netid: MM4:g

```

```

3.86405e-17 netid: MM7:d
2.77537e-18 netid: MM7:d
2.29436e-19 netid: c_43:n
3.66791e-17 netid: MM7:d
3.35031e-18 netid: c_43:n
3.48897e-17 netid: MM5:d
2.77537e-18 netid: MM5:d
3.86405e-17 netid: MM5:d
5.04182e-19 netid: MM3:s
5.69945e-19 netid: MM0:s

```

Then, we generated a new config view for our cell and linked it to the calibre view, which contains the aforementioned netlist with the parasitics. We then set up the old test bench to use this new view and ran the automatic simulations needed to compile the Liberty file and compare with the schematic simulations.

In order to fill in the Liberty file faster we developed a Matlab script that convert the traces exported in .csv from Virtuoso into the correct format for the .lib file, which is shown in the following snippet:

```

/*****
Module           : INV_X4
Cell Description : Combinational cell (INV_X4) with drive strength X4
*****/
cell (INV_X4) {
drive_strength    : 4;
area              : 1.0975; //1.330000 (2.930325um2 vs 2.5632um2);
pg_pin(VDD) {
voltage_name : VDD;
pg_type      : primary_power;
}
pg_pin(VSS) {
voltage_name : VSS;
pg_type      : primary_ground;
}
cell_leakage_power : 57.412850;
leakage_power () {
when      : "!A";
value     : 40.409160;
}
leakage_power () {
when      : "A";
value     : 74.416540;
}
pin (A) {
direction : input;
related_power_pin : "VDD";
related_ground_pin : "VSS";
capacitance : 6.258425;
fall_capacitance : 5.700054;
rise_capacitance : 6.258425;
}
pin (ZN) {
direction : output;
related_power_pin : "VDD";
related_ground_pin : "VSS";
max_capacitance : 242.920000;
function : "!A";
timing () {
related_pin : "A";
timing_sense : negative_unate;
cell_fall(Timing_7_7) {
index_1 ("0.00117378,0.00472397,0.0171859,0.0409838,0.0780596,0.130081,0.198535");
index_2 ("0.365616,1.893040,3.786090,7.572170,15.144300,30.288700,60.577400");

```

```

values ("0.00403093,0.00453635,0.00649363,0.00908681,0.01205778,0.01558255,0.01967407", \
        "0.00403093,0.00447980,0.00611888,0.00822323,0.01245318,0.00453635,0.00498312", \
        "0.00658711,0.00868960,0.01293447,0.00649363,0.00703070,0.00885536,0.01094698", \
        "0.01498347,0.00908681,0.00978884,0.01208531,0.01483173,0.01956196,0.01205778", \
        "0.01295247,0.01588381,0.01918513,0.02490650,0.01558255,0.01666075,0.02020571", \
        "0.02437700,0.03099451,0.01967407,0.02096638,0.02523028,0.02971917,0.03781962", \
        "0.02061456,0.02119501,0.02320684,0.02764001,0.03473818,0.04236789,0.05051590");
}
cell_rise(Timing_7_7) {
index_1 ("0.00117378,0.00472397,0.0171859,0.0409838,0.0780596,0.130081,0.198535");
index_2 ("0.365616,1.893040,3.786090,7.572170,15.144300,30.288700,60.577400");
values ("0.00351845,0.00372798,0.00441003,0.00475651,0.00444414,0.00336489,0.00146739", \
        "0.00351845,0.00391452,0.00529742,0.00707787,0.01054752,0.00372798,0.00410902", \
        "0.00546744,0.00722819,0.01070753,0.00441003,0.00489747,0.00658353,0.00847854", \
        "0.01182217,0.00475651,0.00539066,0.00746919,0.00989820,0.01411477,0.00444414", \
        "0.00524778,0.00783277,0.01081240,0.01595014,0.00336489,0.00432378,0.00752059", \
        "0.01109738,0.01705111,0.00146739,0.00258197,0.00626809,0.01054894,0.01757805", \
        "0.01749551,0.01757651,0.01861294,0.02121072,0.02434296,0.02707745,0.02886522");
}
fall_transition(Timing_7_7) {
index_1 ("0.00117378,0.00472397,0.0171859,0.0409838,0.0780596,0.130081,0.198535");
index_2 ("0.365616,1.893040,3.786090,7.572170,15.144300,30.288700,60.577400");
values ("0.00162617,0.00161542,0.00252926,0.00418593,0.00638010,0.00897348,0.01205494", \
        "0.00162617,0.00192743,0.00309342,0.00463571,0.00792159,0.00161542,0.00192664", \
        "0.00308074,0.00462859,0.00791022,0.00252926,0.00275077,0.00361606,0.00481668", \
        "0.00777997,0.00418593,0.00448144,0.00538881,0.00682993,0.00904189,0.00638010", \
        "0.00674012,0.00801811,0.00939580,0.01185202,0.00897348,0.00943400,0.01095045", \
        "0.01273934,0.01538964,0.01205494,0.01256300,0.01437403,0.01667493,0.01973136", \
        "0.01428300,0.01432561,0.01415400,0.01422246,0.01694084,0.02072039,0.02565933");
}
rise_transition(Timing_7_7) {
index_1 ("0.00117378,0.00472397,0.0171859,0.0409838,0.0780596,0.130081,0.198535");
index_2 ("0.365616,1.893040,3.786090,7.572170,15.144300,30.288700,60.577400");
values ("0.00161845,0.00167783,0.00289582,0.00471797,0.00712435,0.01010073,0.01378325", \
        "0.00161845,0.00192487,0.00304051,0.00455216,0.00747016,0.00167783,0.00195035", \
        "0.00303092,0.00452708,0.00763549,0.00289582,0.00313963,0.00397009,0.00499015", \
        "0.00752167,0.00471797,0.00503286,0.00616629,0.00736290,0.00973715,0.00712435", \
        "0.00766871,0.00906592,0.01058132,0.01316659,0.01010073,0.01058068,0.01213061", \
        "0.01418079,0.01740106,0.01378325,0.01433846,0.01618735,0.01876466,0.02216296", \
        "0.01381037,0.01346539,0.01361817,0.01414654,0.01811789,0.02255024,0.02901449");
}
}
internal_power () {
related_pin      : "A";
fall_power(Power_7_7) {
index_1 ("0.00117378,0.00472397,0.0171859,0.0409838,0.0780596,0.130081,0.198535");
index_2 ("0.365616,7.591250,15.182500,30.365000,60.730000,121.460000,242.920000");
values ("-0.000135,-0.000209,-0.000286,-0.000438,-0.000743,-0.001351,-0.002568", \
        "-0.000338,-0.000411,-0.000488,-0.000641,-0.000945,-0.001554,-0.002771", \
        "-0.001009,-0.001117,-0.001206,-0.001357,-0.001660,-0.002268,-0.003484", \
        "1.646543,0.958443,0.492551,-0.002713,-0.003034,-0.003637,-0.004850", \
        "4.381760,3.649700,2.962753,1.995582,0.998205,0.299721,-0.006985", \
        "7.982441,7.414641,6.730594,5.421472,3.680040,2.046783,0.995845", \
        "12.729920,12.330210,11.654350,10.290670,7.944608,5.204591,2.927285");
}
rise_power(Power_7_7) {
index_1 ("0.00117378,0.00472397,0.0171859,0.0409838,0.0780596,0.130081,0.198535");
index_2 ("0.365616,7.591250,15.182500,30.365000,60.730000,121.460000,242.920000");
values ("7.027624,7.399411,7.479360,7.555385,7.644741,7.187666,6.888935", \
        "6.872874,7.018571,7.147404,7.158635,7.448545,7.411758,7.678863", \
        "7.197088,7.456530,7.318267,7.372508,7.384601,7.467069,7.149943", \
        "8.429678,8.497916,8.790671,8.413163,7.822837,7.284865,6.837121", \
        "11.509250,10.945040,10.789670,10.683480,9.813967,9.068089,8.488414", \

```

```

"16.241270,15.427200,14.843280,13.883830,12.980790,11.408180,9.848604", \
"22.585720,21.617980,20.697690,19.338640,17.575730,16.011860,13.461310");
}
}
}
}

```

We also developed another Matlab script to plot the results of the simulations, shown in the following figures.

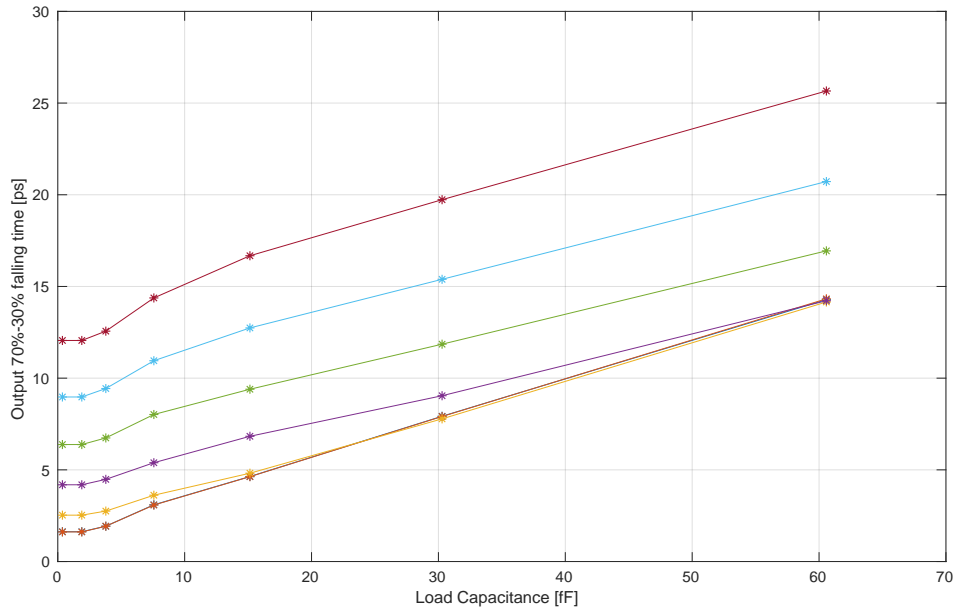


Figure 5: Fall time vs load capacitance, higher curves are for longer input rise times.

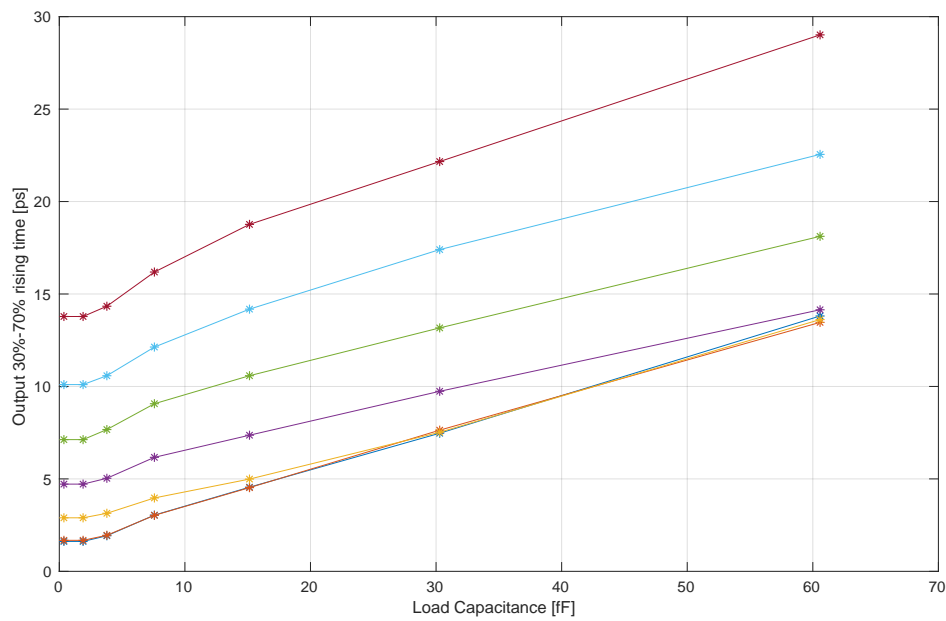


Figure 6: Rise time vs load capacitance, higher curves are for longer input fall times.

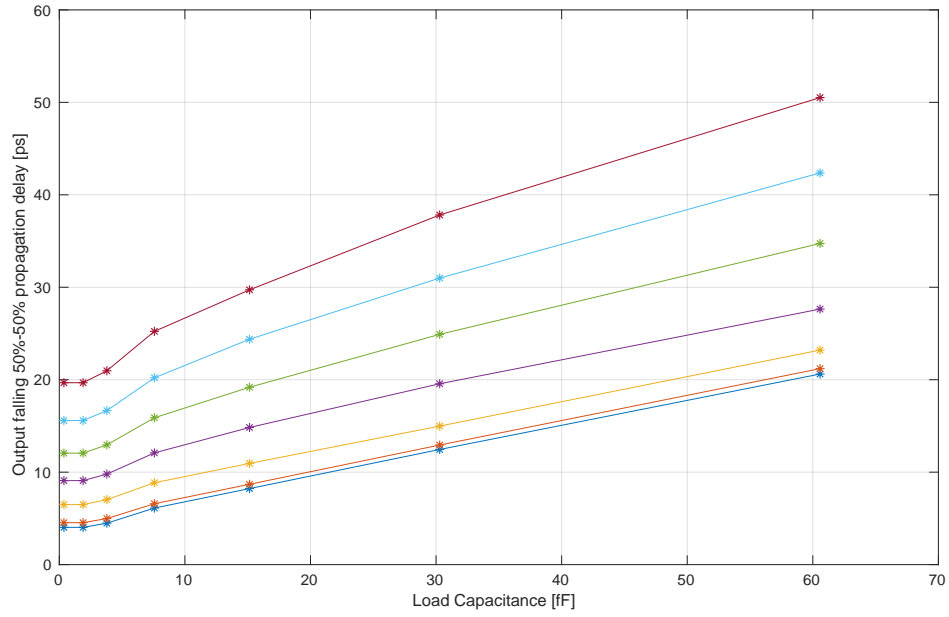


Figure 7: Falling propagation delay vs load capacitance, higher curves are for longer input rise times.

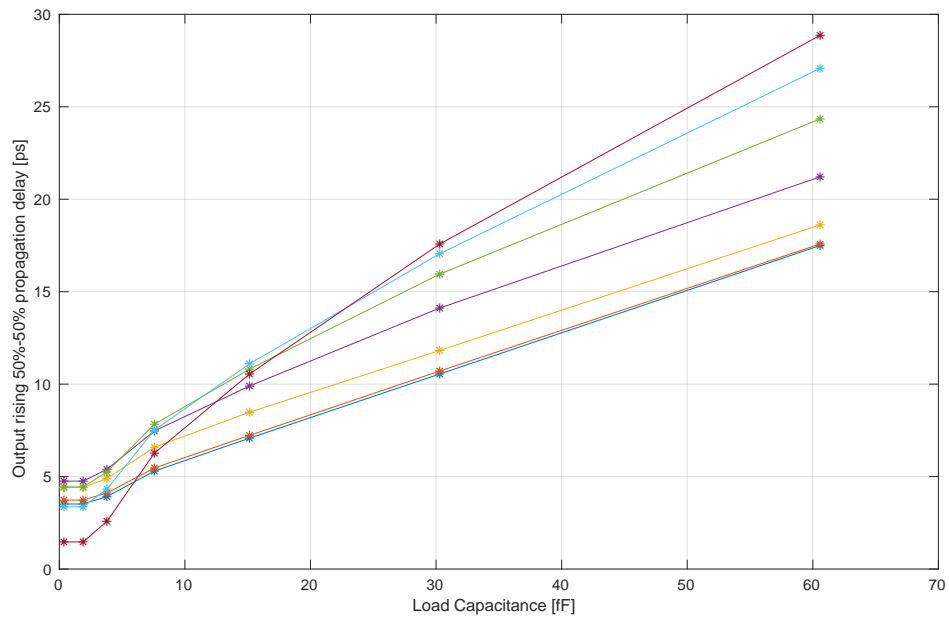


Figure 8: Rising propagation delay vs load capacitance, higher curves are for longer input fall times.

We also compared these results with the simulation done on the original schematic, computing the relative difference in the measured parameters for a given input delay (different input delay values gave similar results and thus were not meaningful to show).

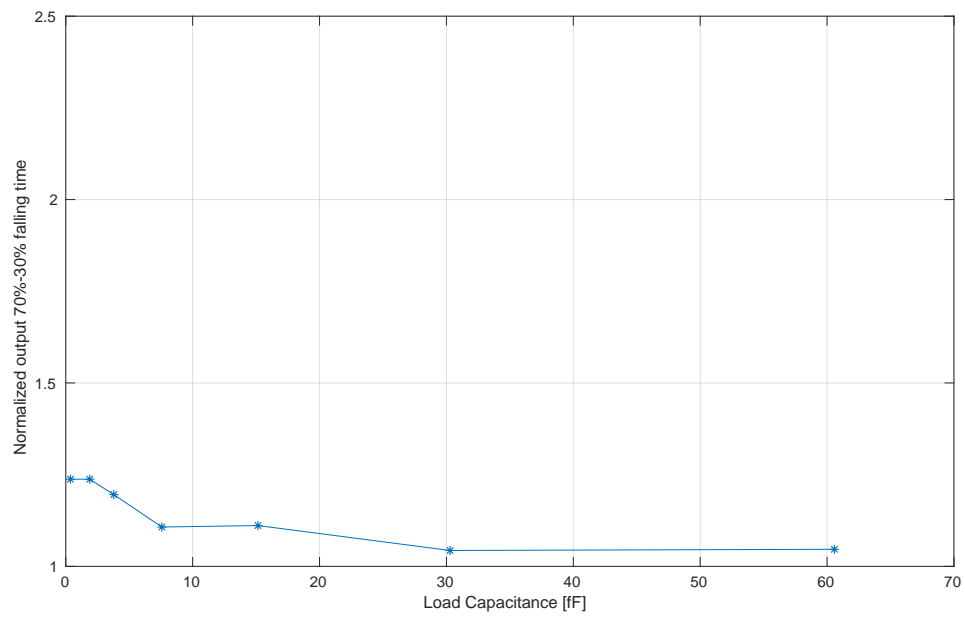


Figure 9: Relative fall time difference vs load capacitance

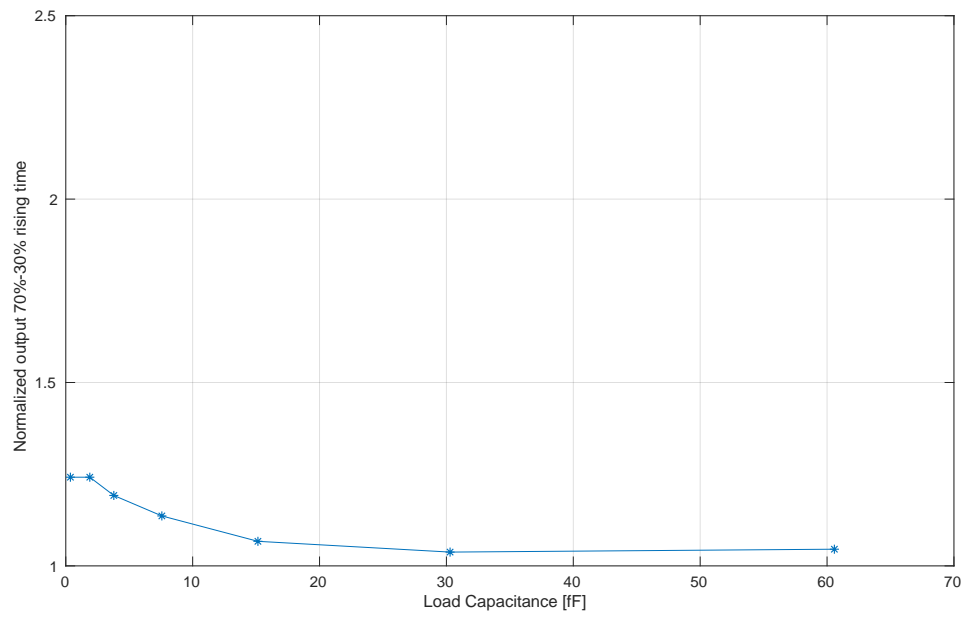


Figure 10: Relative rise time difference vs load capacitance

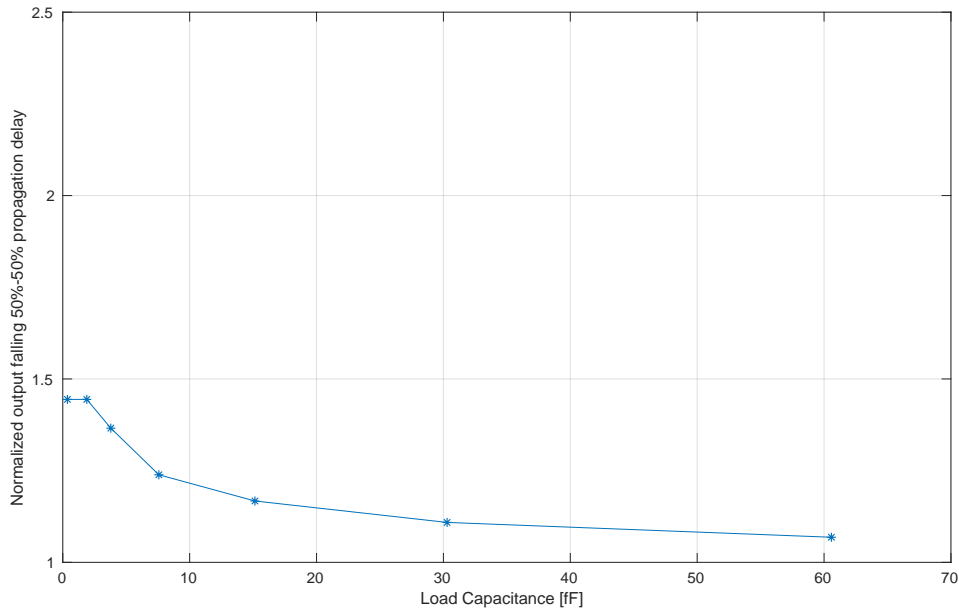


Figure 11: Relative falling propagation delay difference vs load capacitance

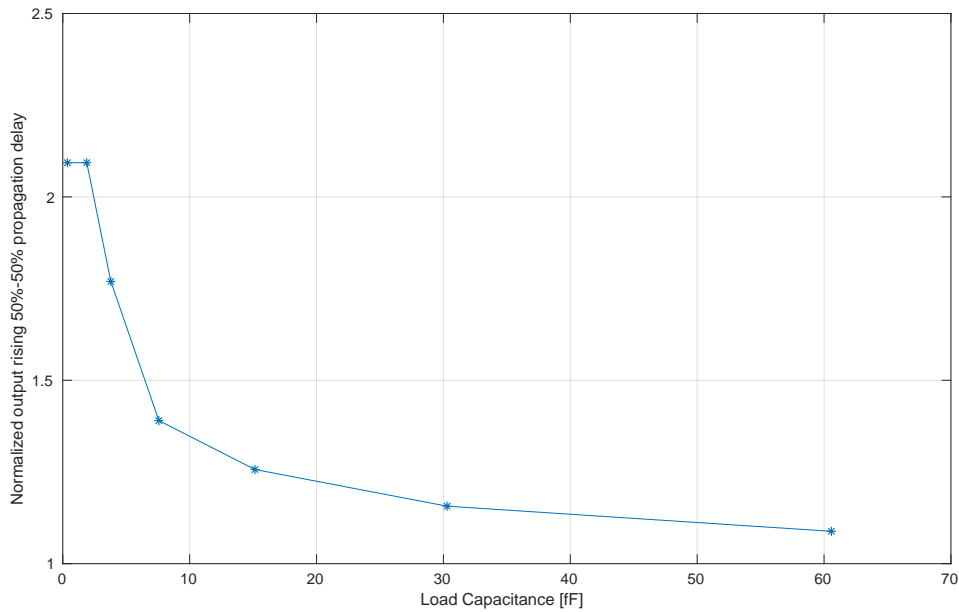


Figure 12: Relative rising propagation delay difference vs load capacitance

We can note that for very small load capacitance values the simulations done on the extracted parasitics show steadily longer delays, from 20% to 45% longer than the schematic-only simulations. This is easily explained by the fact that the parasitic elements have a bigger impact on performance when the load capacitance is very small, while for large values of load capacitance its effect becomes dominant and the weight of the parasitic contributions drastically reduces.

The case of the rising propagation delay is a bit different in that it shows a very large difference between simulations for small load capacitance values: greater than

100%. This behavior has shown to be consistent among different measurements and tests, but we could not really explain why it shows up, as we expected rising and falling delays to be more or less symmetrical.

3 Half adder

3.1 Schematic

The schematic was drawn using Virtuoso Schematic Editor (figure 14), following the reference .png shown in figure 13.

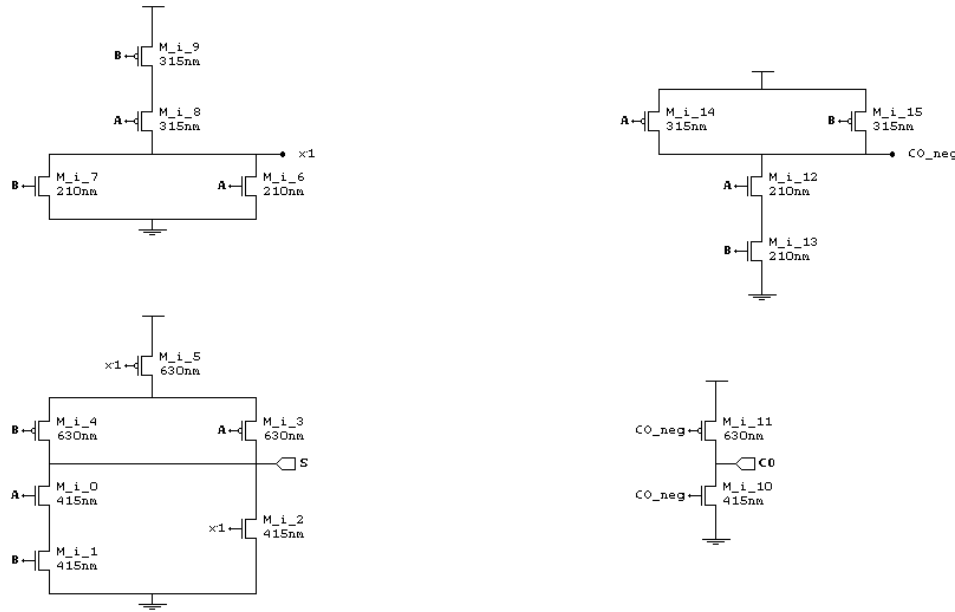


Figure 13: Reference schematic of the half-adder

Even though the 2-1 AOI has the pull up net with a series pMos connected through the power supply and two pMos in parallel, we decided to switch the two subnets to obtain better performances. There are two main reasons for this choice:

1. If A and B are both at '0' and x1 is at '1', the intermediate node is discharged. When a transition of x1 happens then the node has to be charged from only one R_{eq} . With two pMos between the supply and the other transistor the situation could be hypothetically better: in the worst case we could have the same situation as before, but if both A and B are asserted then the capacitance takes less time to complete the charging.
2. The capacitance of the intermediate node is greater if there are 2 pMos before the output instead of one.

We report here the final schematic used to create the layout.

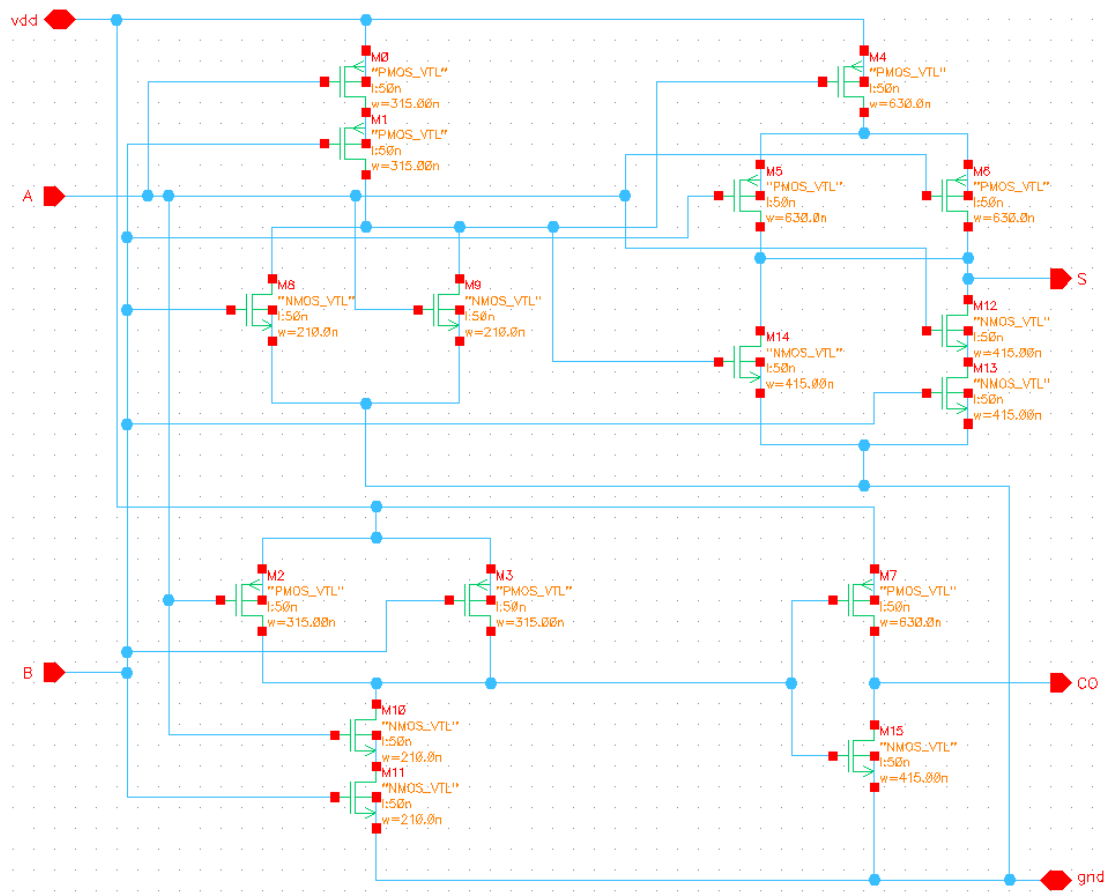


Figure 14: HAX1 schematic

We created a testbench and simulated the circuit as we did for the inverter. Everything was correct.

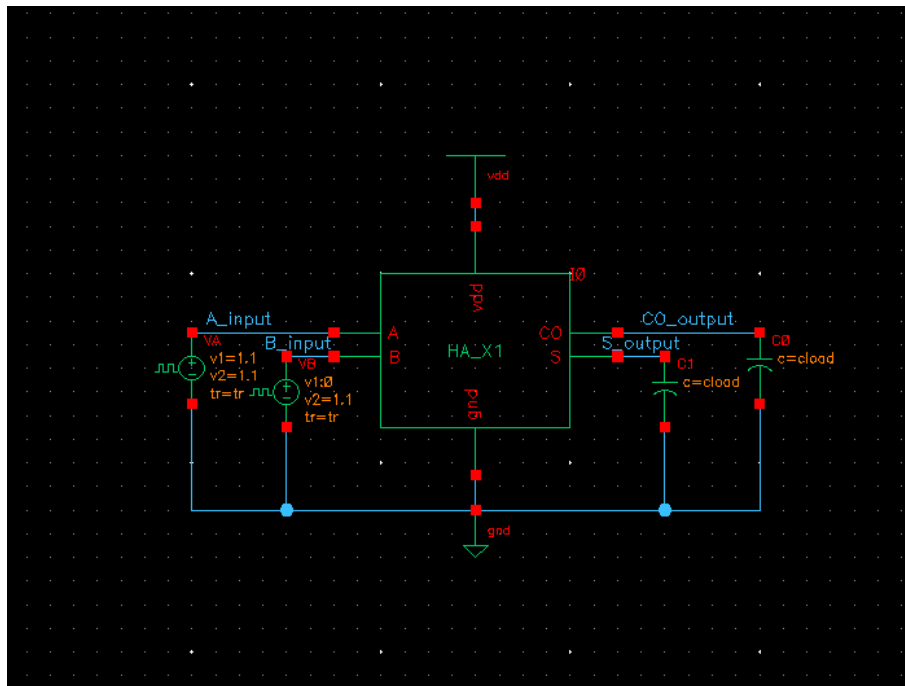


Figure 15: HAX1 testbench

3.2 Layout

Given that the layout of the half adder is quite more complex than the one of the inverter, we started by deeply analyzing the possible approaches using the pen-and-paper method. The final solution that we found consisted in sharing source/drain diffusions as much as possible, in order to minimize the area in the horizontal direction, similarly to what we did in the inverter design.

When actually drawing the design in Virtuoso, we made sure of running the Design Rule Checker (DRC) very often, in order to avoid problems in advance. The final result of this tool is totally similar to the one we obtained for the INV_X4 in section 2.1, and thus omitted.

Layout design started looking at the schematic. First of all we tried to put the two outputs on the opposite sides of the cell, because this way, if the input are brought over it in the middle with the `metal 2` and linked with vias, the input-output paths are more probably balanced in length, keeping the input polysilicon lines shorter and therefore reducing these lines' propagation delays. This assumption lead us to place the carry-out generation network (a NAND gate followed by an inverter) on the left of the layout, and the sum generation network (NOR gate followed by an AOI12 gate) on the right side. Thus we started drawing on a paper the first NMOS of the inverter whose output was CO. With a view to share the sources/drains as much as possible we began putting all the transistors one near the other drawing only the strictly necessary metal. We used different colours to have an easier readability and go on drawing only the `metal` for the sources/drains and the `polysilicon` for the gates until the end of the port and left the routing at last. Then we completed the layout linking together the various elements coherently with the schematic view. The resulting pencil draw is reported in figure 16.

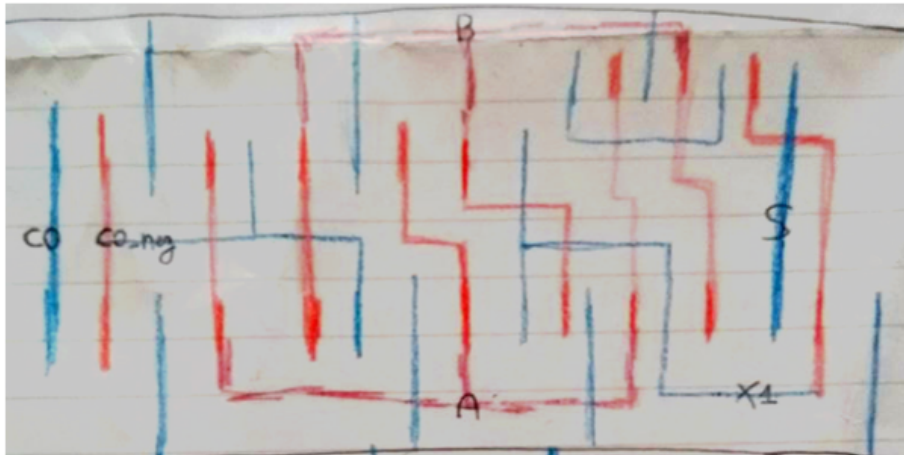


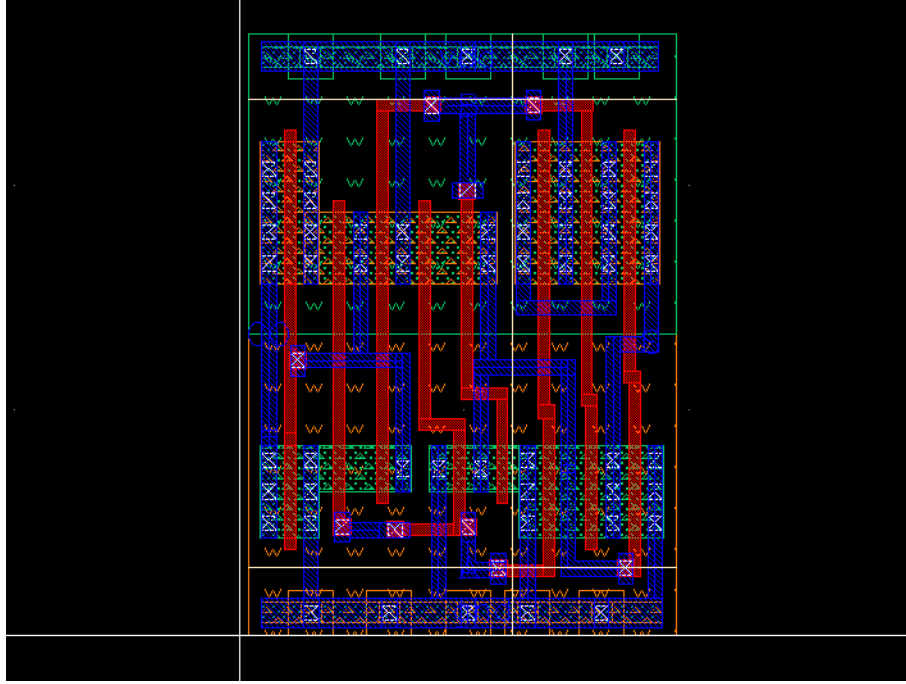
Figure 16: First sketch of the layout. Blue: metal1, red: polysilicon. Labels do not match the position of the corresponding vias.

Then we converted this sketch in a real layout design using the Virtuoso embedded layout editor. During our sessions we paid attention to reduce every distance to the minimum with the aid of the meter tool and in accordance with the Design Rules. We made a lot of effort to have a cell with the minimum area because the area is directly linked to the yield of the process and to logic ports density, capacitance and therefore delay and power cost per commutation. The less is the area, the more ports can stay in a chip or the more the chip will be small, and it is harder to find defects on a single chip if it is small, because the density of the defects per unit

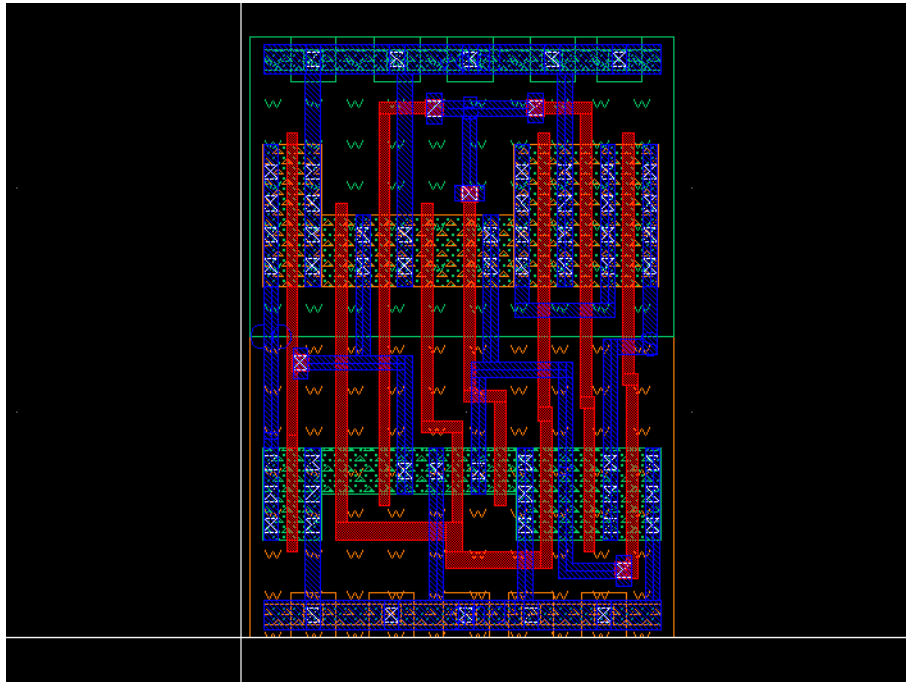
area is about constant (higher yield for the process). The delay and the power costs rise with the area because if more material is used, more resistance/capacitance is added. For this reason we also tried to reduce the length of every wire.

The height of the cell is fixed because of the standard-cell format requirements. Its top and ground lines, whose purpose is to feed the transistors, are made of **metal1**. Many entities like this one can be put one next to the other to form a row of cells packed together and fed with global V_{DD} and GND voltage lines.

When we drew the layout with the editor we changed a little bit the connections. Instead of putting a single wire of **polysilicon** to connect each gate with the related input A/B, we used vias and **metal1** pieces of line. It would have been interesting to compare the performances of two implementations, to see whether longer polysilicon lines perform better or worse than shorter ones connected through metal lines and vias. Both the layouts are reported in figure 17, though only the polysilicon and metal version (figure 17a) has been simulated and characterized due to the lack of time.



(a) Polysilicon and metal input lines



(b) Polysilicon only input lines

Figure 17: Different layouts

Notice how the final layout in figure 17a appears divided in two different active region both for the NMOS and PMOS networks. This is necessary whenever two metal lines are connected to the active region without a gate between them. In fact, during the manufacturing process, the polysilicon layer is lay before the active one. Therefore, the gates prevent the active regions to extend below them, even if the layout masks do. This automatically isolates the drain and source implants and the related metal lines. When no gate is lay between two metal lines, the isolation must be considered as part of the active layer mask. In the early stages of the development

this wasn't clear to us, as proved by figure 17b.

Finally we launched the LVS and then extracted the parasitic elements. The PEX operation gave us a schematic with all the MOS with the relative parasitics. The net is implicit because no wire is drawn, but every terminal has its own reference. The following figures show the group of the 16 MOS of the cell, a detail of one MOS to highlight the labels on the terminals and a parasitic capacitance between the drain of that MOS and gnd.



Figure 18: Detail of the extracted parasitics - group of MOS

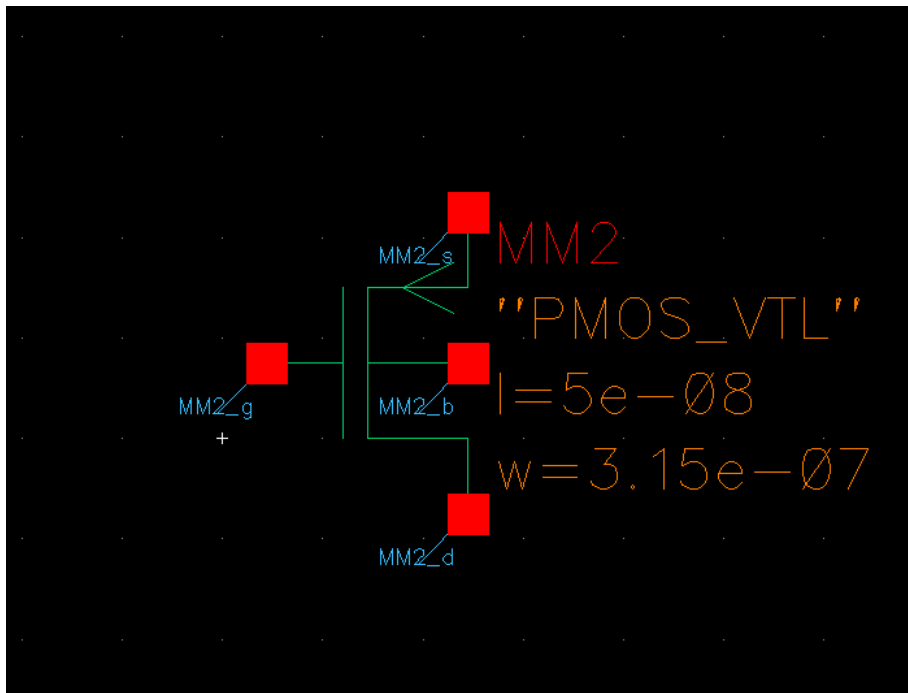


Figure 19: Detail of the extracted parasitics - MOS M2

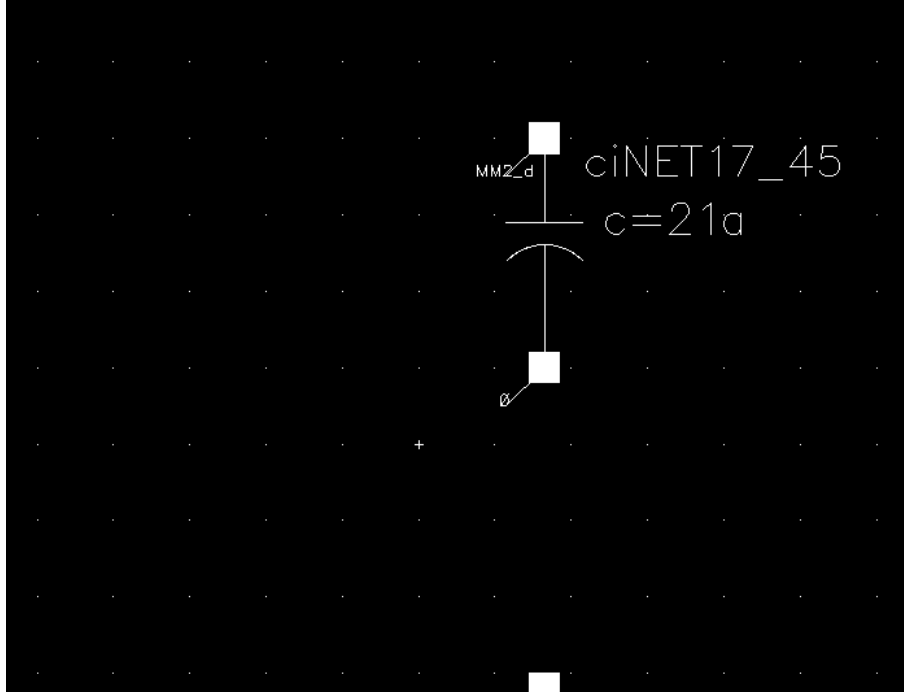


Figure 20: One parasitic capacitance, between the drain of M2 and gnd

Every reference can be spotted looking at the schematic and at the relative file. The PEX produced also the report and the netlist files. In the netlist all the informations about the net with the parasitics can be found. For example here we report the part related with the previous pictures.

```

102
103 mr ni "NET17" 287.286 1.72143e-16 2.2726e-16 '( "MM11_d" "MM2_d" "MM3_d" "MM15_g"
    "MM7_g" )
104 mr_pp 'c "ciNET17_41" ' ("NET17_26" "0") 0.00137581f
105 mr_pp 'c "ciNET17_42" ' ("c_104_n" "0") 0.00658474f
106 mr_pp 'c "ciNET17_43" ' ("c_91_n" "0") 0.0028344f
107 mr_pp 'c "ciNET17_44" ' ("c_102_n" "0") 0.00544915f
108 mr_pp 'c "ciNET17_45" ' ("MM2_d" "0") 0.0209999f
109 mr_pp 'c "ciNET17_46" ' ("c_98_n" "0") 0.00629705f
110 mr_pp 'c "ciNET17_47" ' ("MM11_d" "0") 0.0193463f
111 mr_pp 'c "ciNET17_48" ' ("MM7_g" "0") 0.0617933f
112 mr_pp 'c "ciNET17_49" ' ("MM15_g" "0") 0.0474626f
113 mr_pp 'r "rNET17_50" ' ("c_104_n" "NET17") 3.9
114 mr_pp 'r "rNET17_51" ' ("c_91_n" "c_104_n") 89.3538
115 mr_pp 'r "rNET17_52" ' ("NET17_18" "NET17_26") 0.160909
116 mr_pp 'r "rNET17_53" ' ("c_102_n" "NET17_19") 0.212317
117 mr_pp 'r "rNET17_54" ' ("c_102_n" "NET17_18") 0.716154
118 mr_pp 'r "rNET17_55" ' ("NET17_13" "NET17_26") 0.0418175

```

Figure 21: Part of the file 'HA-X1.pex.netlist'

3.3 Characterization

As done before, we proceeded with the lumped parameters extraction from the designed layout using the PEX tool. Again, the output log and schematic listed all the extracted resistances and capacitances similarly to what has been reported for the INV_X4 in section 2.3. From the extracted data we created a new cell configuration (config view) for the HA_X1. At this point we created a new testbench cell, where we imported our HA_X1 as DUT. Two pulse generators and two load capacitors with parametric values completed the testbench along with the V_{DD} and GND ports.

At this point we were ready to perform the simulations required to fill the timing characterization tables in the Liberty file.

- **Carry-out** Since the C0 generation network is an AND gate of the two inputs A

and B, it is necessary to have one of these inputs to be fixed at its high value for the output to switch. Therefore only eight measurements are required to fully characterize this output pin: output falling and rising propagation delays, output fall and rise time, each of these related to both A and B inputs.

- **Sum** The S generation network is a XOR gate. Its output can thus switch whenever the inputs change from being equal to being different and viceversa. Therefore sixteen measurements are required. It is important to notice that unlike C0, this output shows an inverting behaviour with respect to the related input when the other input is high.

Having set properly the measurement parameters in the ADE L simulation environment we ran the parametric simulation with the same input transition times and load capacitance values used for the INV_X4.