Process corners

Suppose you size a resistor to 100 000.0 Ω, and the simulator agrees with you to seven digits. The resistor that comes back from the fab will be somewhere between 87.5 kΩ and 112.5 kΩ, and nobody, SkyWater included, can tell you where in that range until the wafer has been measured.

You might take that for a defect, but it's the normal, documented behaviour of a semiconductor process, and the PDK ships the numbers so that you can design around them. This page is about how large the variation is, and the next page is about the technique that makes it manageable.

Global variation and local mismatch

A drawn value can be wrong in two different ways. Both look like "error", and people often confuse them.

Global variation, also called **process corners.** Perhaps the furnace ran three degrees warm, the implant dose was at the top of its window, or the poly etch was slightly aggressive. Changes like these shift **every device on the wafer in the same direction, by roughly the same amount**. If your 100 kΩ became 112 kΩ, the 10 kΩ next to it moved by the same proportion.

Local mismatch. Dopant atoms land at random, so two resistors drawn identically, five microns apart on the same die, still differ. A 1 µm × 1 µm strip contains a countable number of impurity atoms, and the count isn't the same in both. This variation is uncorrelated, which means it's different for every pair, and no amount of process control removes it, because it comes from thermodynamics.

You handle corners with design margin, and mismatch with area and layout. If you confuse the two, you get circuits that pass every corner in simulation and then fail on silicon.

The corner files

SKY130 ships the corners as plain parameter files. `libs.tech/ngspice/r+c/` contains `res_low__cap_low.spice`, `res_typical__cap_typical.spice` and `res_high__cap_high.spice`, and `sky130.lib.spice` wires them into library sections named `ll`, `tt` and `hh`. Here are the sheet resistances from the three files side by side, in Ω/□ except where the row says otherwise:

layer	`ll`	`tt`	`hh`	span
n+ diffusion (`rdn`)	111.6	120	128.4	±7.0 %
p+ diffusion (`rdp`)	175.3	197	218.7	±11.0 %
poly (`rp1`)	44	48.2	53.52	−8.7 / +11.0 %
n-well (`rnw`)	1378	1700	2022	±18.9 %
met1 (`rm1`)	0.111	0.125	0.139	−11.2 / +11.2 %
poly contact (`rcp1`, Ω each)	61.28	145.28	213.88	−58 / +47 %

For the precision poly resistor, the corner files set a single explicit knob:

```
+ sky130_fd_pr__res_high_po__var = -0.125      (res_low__cap_low__lin.spice)
+ sky130_fd_pr__res_high_po__var = 0.0         (res_typical__cap_typical__lin.spice)
+ sky130_fd_pr__res_high_po__var = 0.125       (res_high__cap_high__lin.spice)
```

That is **±12.5 %**, written down by the foundry as a multiplier on `rsheet`. The MIM capacitor gets the same treatment: `camimc` is `1.778e-15`, `2.00e-15` and `2.231e-15` F/µm², or −11.1 % / +11.6 %.

The same files also hold a second mechanism, a list of *width* tolerances:

```
+ tol_poly = 0.0287u
+ tol_nfom = 0.0483u
+ tol_m1   = 0.0175u
+ tol_m5   = 0.119u
```

This means your poly can come out 28.7 nm narrower or wider than you drew it. On a 1 µm-wide resistor that's ±2.9 % of the width. What a resistor costs in area recommended 0.35 µm, because area goes as W^2, and on a resistor that narrow the same 28.7 nm is ±8.2 %. That gives you your first real design trade-off: a narrow resistor is cheap in area, but it's also less accurate.

One circuit at three corners

`labs/passives-decks/spice/corners.spice.in` describes a single circuit. Its Makefile writes `corners_tt.spice`, `corners_ll.spice` and `corners_hh.spice` from it, so the three runs differ only in one word on the `.lib` line. Together they print:

```
--- tt : absolute (moves) ---
r_a = 3.550443e+03
r_b = 1.306704e+04
--- ll : absolute (moves) ---
r_a = 3.106637e+03
r_b = 1.143366e+04
--- hh : absolute (moves) ---
r_a = 3.994248e+03
r_b = 1.470042e+04
```

	ll	tt	hh
R_A (drawn 1 × 10 µm)	3106.637 Ω	3550.443 Ω	3994.248 Ω
relative to typical	**0.875000**	1.000000	**1.125000**

The resistor moves by exactly ∓12.5 %, because that's exactly what the corner file told it to do. The MIM capacitor in the same deck gives:

```
--- ll : MIM capacitance, farads ---   c_mim = 1.797322e-13
--- tt : MIM capacitance, farads ---   c_mim = 2.065822e-13
--- hh : MIM capacitance, farads ---   c_mim = 2.338667e-13
```

That's 179.7322 fF, 206.5822 fF and 233.8667 fF, or 0.87003× to 1.13208× the typical value. You'll run this deck in Lab 04, Part 1.

R and C errors together

The same deck also builds a low-pass filter from one resistor and one capacitor, and prints its corner frequency at each corner:

```
--- ll : RC corner frequency, Hz ---   f3db =  2.82935e+08
--- tt : RC corner frequency, Hz ---   f3db =  2.15410e+08
--- hh : RC corner frequency, Hz ---   f3db =  1.69052e+08
```

corner	$f_{-3\ \mathrm{dB}}$	relative
`ll`	282.935 MHz	1.3135
`tt`	215.410 MHz	1.0000
`hh`	169.052 MHz	0.7848

The filter's corner frequency spans 169 MHz to 283 MHz, a 1.67 : 1 range. The corner frequency is $f = 1/2\pi RC$, and R and C move together (the low corner is low R *and* low C, by construction), so their errors multiply instead of partly cancelling. You can check this by hand: $1.125 \times 1.13207 = 1.2736$, and $1/1.2736 = 0.7852$, against the simulated `hh` ratio of 0.7848. Two ±12 % errors, one in R and one in C, multiply to shift the answer by −21.5 %, and the arithmetic agrees with the simulation to three decimal places.

This is why nobody builds a precision on-chip filter from a bare R and a bare C. If your specification says "corner at 200 MHz ±5 %", this topology fails it before you've drawn anything, and no amount of careful layout will fix that; you need a different topology.

Local mismatch

Everything above applies one number to the whole wafer. When you turn on the *local* mismatch models as well, even two resistors on the same die disagree with each other. `labs/passives-decks/spice/mismatch.spice` runs 200 Monte Carlo samples and prints:

```
  Monte Carlo: 200 runs, seed 12345
  ok       sigma/mean A (1x1)       2.7930 %   (reference  2.7930 %)
  ok       sigma/mean B (1x1)       2.8775 %   (reference  2.8775 %)
  ok       sigma/mean C (10x10)     0.3045 %   (reference  0.3045 %)
  ok       sigma/mean D (10x10)     0.3479 %   (reference  0.3479 %)
```

(It also prints two ratio rows and an area line, which Matching and Pelgrom's law uses.)

A and B are drawn identically, 1 µm × 1 µm, in the same simulation on the same wafer, and yet each carries about 2.8 % of its own random variation on top of whatever the corner did. Lab 04: Two resistors that disagree is about that number.

A rule for specifications

> **Never write an absolute on-chip passive value into a specification you have to meet.** Assume ±20 % on any single resistance or capacitance and ±30 % on any product of the two. If the design doesn't survive that, it needs to stop depending on the absolute value.

If absolute values are this poor, you might wonder how anyone builds a 12-bit ADC, whose job is to be accurate to one part in four thousand. The answer is in the same three corner runs, and the next page shows you where.

Next: Matching and Pelgrom's law.