Lab 01: A resistor you designed

You'll measure the sheet resistance and end resistance of a SKY130 poly resistor from your own simulation, then use those two numbers to size two resistors that land within 1 % of their targets. Along the way you'll work out what a resistance costs in silicon compared with a piece of digital logic.

Package: `labs/lab-01-a-resistor-you-designed/`

Before you start

- Read Getting started, Resistors on a chip, Ohms per square and What a resistor costs in area.
- You need the workbench from IC101: the noVNC desktop comes up in your browser and the IC101 smoke test passes. Part 1 loads this course into it.
- The only circuit theory you need is Ohm's law and a voltage divider.

Parts 1 to 7 are required. Everything under Going further, at the end, is optional.

Time: about 2.5 hours, not counting the optional parts. Each ngspice run is silent for anywhere from about 10 seconds to just over a minute before it prints anything, depending on your machine (Warnings and delays).

Part 1: Load the course

Can you reach the AD102 files, on the tool versions this course's printed outputs come from? If you haven't set up the workbench yet, do IC101 Lab 03 first.

1. On your laptop, from your `workspace` clone, start the desktop (`./scripts/start_vnc.sh`, or `scripts\start_vnc.bat` on Windows), open http://localhost/ and open a terminal on the desktop. Type every other command in this course in that terminal, not on your laptop.

2. Load the course environment and get the course files. The first time, type all four lines; after that, `mod ad102` on its own takes you back to the course.

```
. /foss/designs/common/.designinit    # a new terminal usually does this for you; again
is harmless
echo "PDK is: $PDK"
mod add ad102                         # first time only: clones github.com/uoftasic/ad102
mod ad102                             # every time after that
```

   You should see the `ASIC-EDU: PDK=sky130A ...` banner, `PDK is: sky130A`, then `OK  module ready: /foss/designs/modules/ad102` from `mod add`, then `cwd: /foss/designs/modules/ad102` from `mod ad102`. `mod` and the course files have to work; if they don't, ask for help before going on. If `$PDK` says something other than `sky130A`, ask too, but carry on: every lab deck names its model files by absolute path, so `make` works whatever `$PDK` says. Only XSchem, in the optional last part of Lab 03, reads it (Getting started explains).

If you see	It means
`bash: mod: command not found`	This terminal has not loaded the course environment. Run the `.designinit` line again in it.
`mod: no such module: ad102`	You ran `mod ad102` before `mod add ad102`.
`fatal: could not read Username for 'https://github.com'`	git couldn't read the repo anonymously and is asking you to log in. Check that you typed `ad102`. If you did, ask in Discord. Do not type your password.

   For anything else, such as a blank desktop page, see IC101's Troubleshooting page.

3. Check the two tool versions this course's numbers were recorded with:

```
ngspice -v | head -3
xschem -v | head -1
```

   You should see `** ngspice-47 : Circuit level simulation program` on the second line and `XSCHEM V3.4.8RC`. If you see different versions, you are on a different image, and commands on these pages may fail in ways this course does not describe; ask in Discord before you go on.

4. Go to this lab's folder and list the packages:

```
cd /foss/designs/modules/ad102/labs/lab-01-a-resistor-you-designed
ls ..
```

You should see: `ls ..` lists the four lab packages plus `passives-decks`:

```
lab-01-a-resistor-you-designed     lab-03-a-time-constant-in-silicon  passives-decks
lab-02-how-big-is-a-picofarad      lab-04-two-resistors-that-disagree
```

Part 2: Measure the material

Does the one number on the model card predict what a strip of `res_high_po` measures?

`spice/sheet.spice` builds twenty-two strips of poly, forces exactly 1 µA through each, and prints the voltage across it. Dividing that voltage by 1 µA gives the resistance, so each number is one application of Ohm's law to one device, with no curve fitting. The first block is a **length ladder** of `sky130_fd_pr__res_high_po`, with W fixed at 1 µm and L going from 1 µm to 100 µm.

1. Read the one number the foundry advertises on the model card:

```
grep                  rsheet                  /foss/pdks/sky130A/libs.ref/sky130_fd_pr/
spice/sky130_fd_pr__res_high_po.model.spice
```

```
+ rsheet = 317.3885
+ rsh = {rsheet*(1+sky130_fd_pr__res_high_po__var)*(1+res_match)}
```

   The first line is the number; the second shows how the model uses it.

2. On paper, predict what the seven strips in the length ladder will measure. At $W = 1$ µm, L/W is just L, so you are predicting 317.3885×1, $\times 2$, $\times 5$, $\times 10$, $\times 20$, $\times 50$ and $\times 100$.

3. Run the lab:

```
make
```

   This runs two ngspice decks, `spice/sheet.spice` and `spice/my_resistor.spice`, and then the checker script, `src/check_res.py`. It ends in `FAIL`, which you fix in Part 7. Leave the output in your terminal, because Parts 2 to 5 read from it. If it scrolls away, `grep -E '^---|^r_' results/sheet.log` prints the deck's blocks again, and `make check` prints the checker's part.

4. Find block A in the output and compare it with your predictions:

```
--- A. length ladder, W = 1 um (ohms) ---
r_l1 = 6.954646e+02
r_l2 = 1.012684e+03
r_l5 = 1.964344e+03
r_l10 = 3.550443e+03
r_l20 = 6.722641e+03
r_l50 = 1.623923e+04
r_l100 = 3.210022e+04
```

How far off are your predictions for the longest and the shortest strip, and what does the difference tell you?

Answer: A1, in Answers at the end.

5. Each extra micrometre of length adds the same resistance (for example, $(1012.684 - 695.4646)/1$ and $(1964.344 - 1012.684)/3$ both come to 317.22 Ω), so R is a straight line in L. The line doesn't pass through the origin, though, so you need to write it with an intercept:

$$R = R_{\text{end}} + R_{\square}\,\frac{L}{W}$$

Use the two extremes, $L = 1$ and $L = 100$, to solve for both unknowns. Write them down.

6. Check your line against two measurements you did not use, at $L = 10$ and $L = 20$.

You should see:

$$R_{\square} = \frac{32100.22 - 695.4646}{100 - 1} = 317.2198\Omega/\square \qquad R_{\text{end}} = 695.4646 - 317.2198 = 378.2448\Omega$$

At $L = 10$ the line gives $378.2448 + 3172.198 = 3550.4428$, against a measured 3550.443; at $L = 20$ it gives 6722.6408, against 6722.641. It matches every point to the last printed digit, across a 100-to-1 range of lengths. The checker prints the same extraction, with the course's own reference run in the last column (the `xhigh_po` row is the other material, which you extract yourself in Part 5):

```
the two numbers that define the device
  device          ohm/square    end ohms                 reference
  high_po           317.2198    378.2448      317.2198 / 378.2448
  xhigh_po         2118.7619     34.2111      2118.7619 / 34.2111
```

The 378 Ω is a fixed overhead on every resistor, the same whatever length you draw; Part 6 finds out where it comes from. What fraction of a 100 kΩ resistor is it, and of a 500 Ω one?

Answer: A2, in Answers at the end.

Part 3: Size a 10 kΩ resistor

How much does leaving out the end resistance cost on a real design?

`spice/sheet.spice` already contains two 10 kΩ designs at $W = 1$ µm (block B).

1. Work out the naive length, using only the sheet resistance on the card: $L = 10000/317.3885 = 31.5074$ µm. This is `r_naive` in the deck.
2. Work out the corrected length, $L = (10000 - R_{\text{end}})/R_{\square}$, from the two numbers you extracted, and write it down. The deck's version of this design is `r_fixed`.

3. Put the naive length into your line from Part 2 and predict how far from 10 kΩ it lands. Then find block B in the output.

You should see:

```
--- B. the 10 kohm design, naive vs fixed (ohms) ---
r_naive = 1.037302e+04
r_fixed = 9.999997e+03
```

The naive design is 10,373 Ω, 3.7 % too big. The deck draws `r_fixed` at 30.3315 µm (check your length against it), and it lands at 9999.997 Ω, within three parts in ten million of the target.

An error of 3.7 % sounds small, so why is it worth fixing?

Answer: A3, in Answers at the end.

Part 4: Change the width

Is ohms per square a property of the material alone, and what does a narrow strip buy you?

The length ladder was all at $W = 1$ µm. Blocks C and C2 of the `make` output from Part 2 sweep four other widths at two lengths, $L = 10$ µm and $L = 5$ µm:

```
--- C. width ladder, L = 10 um (ohms) ---
r_w0p35 = 1.067497e+04
r_w0p69 = 5.123000e+03
r_w2 = 1.785180e+03
r_w5 = 7.167730e+02
--- C2. the same four widths at L = 5 um (ohms) ---
r_v0p35 = 5.816766e+03
r_v0p69 = 2.823272e+03
r_v2 = 9.925276e+02
r_v5 = 3.998070e+02
```

1. For each width, fit a slope (Ω per µm of length) and an intercept (end Ω) from the two lengths, the same way you did in Part 2. That is two subtractions per width.
2. Multiply each slope by its width to get Ω per square.
3. For each width, work out the body area that 1 kΩ needs: $W \times 1000/(\Omega \text{ per } \mu\text{m})$. That is the last column of the table below.

You should see: (the $W = 1$ µm row comes from the length ladder in Part 2)

W (µm)	Ω per µm of length	end Ω	Ω per square	µm² of body per kΩ
0.35	971.641	958.562	340.074	0.3602
0.69	459.946	523.544	317.362	1.5002
1	317.220	378.245	317.220	3.1524
2	158.530	199.875	317.061	12.6159
5	63.393	82.841	316.966	78.8728

Read the Ω-per-square column from the bottom up. Where does it stop being constant, and why?

Answer: A4, in Answers at the end.

Now read the last column: 0.36 µm² per kΩ at the narrowest width against 78.9 at the widest, a factor of 219 for the same material. Why is that so much more than the ratio of the widths?

Answer: A5, in Answers at the end.

Part 5: Change the recipe

What does the other poly implant change, and what does a resistor cost next to a piece of digital logic?

Block D repeats the length ladder on `sky130_fd_pr__res_xhigh_po`, at five of the lengths (1, 2, 10, 50 and 100 µm): the same layer and drawing rules, with a different implant dose.

```
--- D. res_xhigh_po, W = 1 um (ohms) ---
r_x1 = 2.152973e+03
r_x2 = 4.271620e+03
r_x10 = 2.122088e+04
r_x50 = 1.059694e+05
r_x100 = 2.119104e+05
```

1. Extract $R_{\square}$ and R_{end} from block D, using the two extremes as in Part 2.
2. Compare with what its model card advertises:

   ```
   grep                    rsheet                    /foss/pdks/sky130A/libs.ref/sky130_fd_pr/
   spice/sky130_fd_pr__res_xhigh_po__base.model.spice
   ```

   ```
   .param  rsheet = 2000.0
   + rbody = {(l*rsheet+rbody_var)*(1+res_match)/w}
   ```

   As in Part 2, the first line is the number and the second shows how the model uses it.
3. Price your resistors in digital logic. In DD103, a 4-bit ripple-carry adder (twelve SKY130 standard cells that add two nibbles) synthesises to 110.11 µm². At $W = 1$ µm a strip's area is its length in µm². With the formula from Part 3, step 2, $L = (R - R_{\text{end}})/R_{\square}$, work out the length of a 1 kΩ, 10 kΩ and 100 kΩ `res_high_po` and a 100 kΩ `res_xhigh_po`, all at $W = 1$ µm, then divide each area by 110.11.

You should see: the card says `rsheet = 2000.0`. Your extraction says 2118.7619 Ω/□, 5.9 % higher, with only 34.21 Ω of ends. Compared with `res_high_po`, that is 6.7 times the ohms per square and a tenth of the fixed overhead. Your price list should match this one:

you drew	area	in adders
1 kΩ, `res_high_po`, W = 1 µm	1.96 µm²	0.018
10 kΩ, `res_high_po`, W = 1 µm	30.33 µm²	0.28
100 kΩ, `res_high_po`, W = 1 µm	314.05 µm²	2.85
100 kΩ, `res_xhigh_po`, W = 1 µm	47.18 µm²	0.43

The checker prints three of these lengths too, to four decimals, in the block headed `and the design that follows from them`: 30.3315 µm for 10 kΩ, and 314.0465 µm and 47.1812 µm for the two 100 kΩ rows. The same block has rows you don't need here: 4.7036 µm for 10 kΩ of `res_xhigh_po`, and a 2200 Ω row for each material. You size a 2200 Ω resistor yourself in Part 7; work it out from your own two numbers rather than copying that row.

The card says 2000.0 and your simulation says 2118.7619. Which one do you design with, and why?

Answer: A6, in Answers at the end.

The areas above are the strip only. A drawn resistor also needs its two contact heads, spacing to whatever is beside it, and usually a guard ring, so treat them as a lower bound rather than a quote: What a resistor costs in area puts a folded 100 kΩ, with its guard ring and dummies, at two to three times its drawn poly. Even so, at 100 kΩ one of the cheapest things in the analog toolbox outweighs a working arithmetic unit. That is why analog blocks look enormous next to digital ones in a die photograph, and why a chip designer never answers "just add a resistor" straight away.

Part 6: Split a resistor into head and body

Which part of the resistor is the 378 Ω you extracted: the contact head, or the body?

Resistors on a chip and Ohms per square quote a different deck from the one you ran in Part 2: `labs/passives-decks/spice/r_head_and_body.spice`. `sky130_fd_pr__res_high_po` is a subcircuit (`.subckt`), and inside it a node called `rb` sits between the contact head and the resistive body. This deck measures the voltage at that node too, so one measured resistance splits into its two parts.

1. Go to the `passives-decks` package and run the resistor deck:

   ```
   cd ../passives-decks
   make resistor
   ```

   It runs one deck and prints its blocks, and there's no `PASS` or `FAIL` at the end.

2. Before you look at block A, predict: when the strip goes from $L = 1$ µm to $L = 10$ µm, which part changes, the head or the body?

3. Read block A:

   ```
   --- A. one strip, split into head and body (ohms) ---
   r_l1 = 6.954646e+02
   head_l1 = 2.998915e+02
   body_l1 = 3.955731e+02
   r_l10 = 3.550443e+03
   head_l10 = 2.998915e+02
   body_l10 = 3.250551e+03
   ```

4. Work out the slope of the body from `body_l1` and `body_l10`.

5. Go back to the lab folder:

   ```
   cd ../lab-01-a-resistor-you-designed
   ```

You should see: the head is 299.8915 Ω at both lengths, the same to seven digits, because you did not change the contacts. The bodies scale:

$$\frac{3250.551 - 395.5731}{10 - 1} = \frac{2854.978}{9} = 317.2198\ \Omega/\square$$

That's the same sheet resistance you extracted in Part 2, this time from a different deck.

The head is 299.89 Ω, but the end resistance you extracted in Part 2 is 378.2448 Ω. Where are the other 78 Ω?

Answer: A7, in Answers at the end.

Part 7: Size your own resistors

Can you size two resistors from your own two numbers so that each lands within 1 % of its target?

`spice/my_resistor.spice` contains two devices and two `L=` numbers. You edit only those two numbers.

	device	W	target
R1	`sky130_fd_pr__res_high_po`	1 µm	2.2 kΩ
R2	`sky130_fd_pr__res_xhigh_po`	1 µm	50 kΩ

The file ships with the naive lengths, so the check at the end of Part 2's `make` failed:

```
R1  sky130_fd_pr__res_high_po  W=1
  target     2200.0 ohm
  got        2577.1 ohm   (+17.14 %)   FAIL
R2  sky130_fd_pr__res_xhigh_po W=1
  target    50000.0 ohm
  got       53001.1 ohm   (+6.00 %)   FAIL
```

1. R1 came out 17.14 % high. Compare that with the 378 Ω of ends as a fraction of 2.2 kΩ. What does it tell you?

   Answer: A8, in Answers at the end.

2. Work out both lengths from the $R_\square$ and R_{end} you extracted for each device in Parts 2 and 5, and write them down.

3. Compare yours with the checker's. Print its report again (the Part 2 output has probably scrolled away):

   ```
   make check
   ```

   Below the two `FAIL` lines, the checker explains each failure and ends the explanation with the length it would use. Its last lines note that on a fresh clone this `FAIL` is expected and doesn't mean the package is broken, because fixing it is the lab.

4. Edit the two `L=` numbers in `spice/my_resistor.spice` with any of the desktop's editors (`gedit`, `nano` and `vim` are installed; for example `gedit spice/my_resistor.spice &`). Write them as plain microns, with no `u` suffix, to four decimals like the numbers already there. The file marks the two lines to change:

   ```
   X1 r1 0 0 sky130_fd_pr__res_high_po  W=1 L=6.9315
   *                                       ^^^^^^ R1: 2.2 kohm goes here
   X2 r2 0 0 sky130_fd_pr__res_xhigh_po W=1 L=25.0000
   *                                       ^^^^^^^ R2: 50 kohm goes here
   ```

5. Re-run your file and the checker. `make mine` simulates only `spice/my_resistor.spice` and `make check` prints the checker's report again, so you can do both in one command:

   ```
   make mine check
   ```

You should see: both resistors marked `PASS` (each must land within 1 % of its target), then

```
PASS -- every number matches the reference run.
```

Finishing

You're done when `make mine check`, run on the two lengths you worked out in Part 7, prints `PASS -- every number matches the reference run.`

That line needs each of your resistors within 1 % of its target. The checker also compares the resistances from `sheet.spice` with the reference run (within 0.5 %), and the two constants extracted from them (within 1 %). A DC operating point has no randomness in it, so on the pinned image (ngspice-47, `sky130A`, `tt` corner, 27 °C) your numbers match the ones on this page to every digit shown.

Question: on area alone, which material would you use for a 100 kΩ resistor, and what does it cost in adders?

Answer: A9, in Answers at the end.

After your own numbers pass, read `solutions/README.md` for the reference sizing and some open questions.

Submit: nothing for this lab.

Going further (optional)

None of these is in the time estimate, and no later part needs them. What to do next on the Course path is the last line of this page.

End resistance from the model card

Open the model card from Part 2, step 1, `sky130_fd_pr__res_high_po.model.spice`. The body is declared `l = {leff}` with `leff = {l + 0.247}`, so the strip is electrically 0.247 µm longer than you drew it, worth $317.2198 \times 0.247 = 78.35\ \Omega$. The head is a second resistor, `rsh = 345.8312` over a width of `weff + 0.1558 = 1.1548` (`weff` is the effective width, 0.999 µm for a strip drawn 1 µm wide; the first answer in Part 4 has more on it), giving $345.8312/1.1548 = 299.47\ \Omega$. Its temperature coefficients (`tc1 = -4.3e-4`, `tc2 = 12e-6`, `tnom = 30`) scale that by $1 + (-4.3 \times 10^{-4})(-3) + (12 \times 10^{-6})(9) = 1.001398$ at the 27 °C ngspice simulated at ($27 - 30 = -3$, and $(-3)^2 = 9$), giving 299.89 Ω. And $299.89 + 78.35 = 378.24$, against the 378.2448 you measured. The arithmetic closes to five figures using only constants in a text file you can `cat`.

More of the `make resistor` output

Part 6's `make resistor` printed three more blocks.

- Block C is one 314.046460 µm × 1 µm strip (the length that gives 100 kΩ of `res_high_po`) simulated as four different resistor models: the two poly implants, and the p+ and n+ diffusion resistors. The drawing and the silicon area are the same, and the values spread over 18:1:

```
--- C. one 314 um x 1 um strip, four implants (ohms) ---
r_high = 1.000000e+05
r_xhigh = 6.653872e+05
r_pdiff = 6.138915e+04
r_ndiff = 3.689775e+04
```

- Blocks B and D are the worked examples on What a resistor costs in area: 1 k, 10 k and 100 k sized with the design equation, and the same 100 kΩ at four widths. Check that each lands on its target.

Part 8 (optional): The `u` suffix trap

`spice/u_trap.spice` holds the same resistor written twice, once as `W=1 L=10` and once as `W=1u L=10u`. Getting started explains what goes wrong.

1. Run it:

```
make utrap
```

You should see:

```
--- the same device, written two ways (ohms) ---
good = 3.550443e+03
bad = 3.193812e+03

   and the one line that told you, buried in the log:
Warning: r.xb.rbody: resistance too low or not given, set to 1 mOhm
   ngspice exit status: 0
```

The wrong spelling gives 3193.812 Ω instead of 3550.443 Ω, 10 % off, with no error and exit status 0. Apart from `u_trap.spice`, which exists to show the mistake, there is no `u` suffix anywhere in this package, and there should be none in yours.

Part 9 (optional): Draw a resistor in SiliWiz

SiliWiz is a browser layout tool from the Tiny Tapeout project. It shows a cross-section of your drawing while you draw it, which makes "a strip of poly with two contacts" concrete.

1. Open **https://app.siliwiz.com/?preset=blank**.
2. Pick the **polyres** layer from the palette on the left and draw a long thin rectangle.
3. Pick **metal1 via** and drop one square at each end of it.
4. Pick **metal1** and put a contact pad over each via.
5. Click each metal1 pad, choose **Set Label** (or press `S`), and name them `in` and `out`.
6. Tick **Show SPICE** at the bottom. Find the line starting `R0 in out`; that is your resistance.
7. Watch the **cross-section** view as you drag the rectangle longer and thinner.

You should see: a line starting `R0 in out` in the SPICE output, and its value changing as you reshape the strip.

There is a guided version of this exercise at https://tinytapeout.com/siliwiz/resistors/.

> **Warning:** SiliWiz is not SKY130, and its numbers are not the ones in this lab. Its `polyres` layer is documented as *"400 ohms per square"*. SKY130's `res_high_po` is 317.2198 Ω/□ measured, and 319.8 Ω/□ in the Magic layout tool's tables (see Ohms per square). SiliWiz uses round numbers chosen for teaching. Use it for the cross-section and ngspice for the value.

Check the rest of the image

AD102 doesn't use these tools. They are the AD104 layout tools, listed so you can confirm the image is complete:

Tool	Command	Expect
Magic	`magic -dnull -noconsole -- version`	`8.3.681`
KLayout	`klayout -v`	`KLayout 0.30.11`
Netgen	`netgen -batch quit | head -1`	`Netgen 1.5.323 compiled on …`

`netgen -batch quit` also prints `Warning: netgen command 'format' use fully-qualified name '::netgen::format'` and a second line like it. Those are Tcl namespace notices from the tool itself. They have nothing to do with your design, so there's nothing to fix.

Next on the Course path: read the four pages in row 3, starting with Capacitors on a chip, then do Lab 02: How big is a picofarad?.

Answers

A1. Part 2: Measure the material

You predicted 31,739 Ω for the longest strip, and it measured 32,100 Ω, which is 1 % high. You predicted 317 Ω for the shortest, and it measured 695 Ω, a factor of 2.2 higher. The same formula is that close at one end and that far off at the other, so something is being added that doesn't depend on how long you drew the strip.

A model that is right at one end of your design range and wrong at the other is worse than one that is wrong everywhere, because it passes every check you run on the big case.

A2. Part 2: Measure the material

It's 0.4 % of a 100 kΩ resistor, which you can ignore, but it's most of a 500 Ω one.

A3. Part 3: Size a 10 kΩ resistor

Think about what resistors are for. A resistor that sets a bias current sets it 3.7 % wrong. A pair of them setting a gain of 10 gets the gain wrong too, because the fixed 378 Ω is a different fraction of a long resistor and a short one. And the error is systematic: every chip on every wafer gets the same 3.7 %, so no amount of averaging finds it. A random spread is a tolerance, which you budget for; an error that is the same on every chip is a mistake, which you fix.

A4. Part 4: Change the width

From the bottom up the column reads 316.966, 317.061, 317.220 and 317.362, constant to a tenth of a percent, so sheet resistance behaves as a property of the material. That holds until the top row, 0.35 µm, where it jumps 7 % to 340.074. Below 0.69 µm the model applies `weff = w - 0.001 - 0.0672 * (0.69 - w)`, so a strip drawn 0.35 µm wide conducts as though it were 0.326 µm wide, because lithography and etch don't give you exactly the width you drew.

The end resistance changes with width as well, from 82.841 Ω at 5 µm to 958.562 Ω at 0.35 µm, because the contact head is as wide as the strip. Its resistance is `rsh` divided by a width; see End resistance from the model card under Going further.

A5. Part 4: Change the width

You win twice by going narrow, once on width and again on the length you no longer need, so area per ohm goes as W^2. That makes "just draw it narrow" look free, but a narrow strip costs you something in matching, and Lab 04 measures how much.

A6. Part 5: Change the recipe

Design with the simulation. The headline number on a model card is a nominal value, whereas the device that gets fabricated behaves the way the model computes.

The implant is also the only knob in this lab that is not geometry. The foundry gives you a short menu of implant doses, and choosing from it is the one way to change ohms per square instead of buying more area.

A7. Part 6: Split a resistor into head and body

They're in the body. The model's `leff` makes the strip 0.247 µm longer than you drew it, and that extra length is worth 78.35 Ω. The 378.2448 Ω is the 299.89 Ω head plus that 78.35 Ω body extension. End resistance from the model card, under Going further, has the arithmetic.

A8. Part 7: Size your own resistors

378.2448/2200 is 17.2 %, about a sixth, so almost all of R1′s error is the ends that the naive length left out. On a 2.2 kΩ resistor the ends are no longer a rounding error.

A9. Finishing

`res_xhigh_po`: 47.18 µm², 0.43 adders, against 314.05 µm² and 2.85 adders for `res_high_po` (Part 5). Area doesn't settle it, though. `res_xhigh_po`'s model card has a voltage coefficient, `vc1_body = -1.00e-3`: its resistance moves by about 0.1 % per volt across it, which doesn't matter in a bias string but shows up as distortion in a signal path. The lab's `solutions/README.md` has the longer version.