The full DRC rule set

Magic's SKY130 technology file defines its design rules in several **variants**, and it lists them on this line:

```
style drc variants (fast),(full),(routing)
```

The first variant listed is the default, so unless you choose another one, Magic checks your layout with **`fast`**, which leaves out a whole category of rules.

What `full` adds

The `full` variant adds four rules that matter to you, and none of them is in the default set:

```
variants (full)
 cifmaxwidth nwell_missing_tap 0 bend_illegal
        "All nwells must contain metal-connected N+ taps (nwell.4)"
 cifmaxwidth tap_missing_licon 0 bend_illegal
        "All taps must be contacted (licon.16)"
 cifmaxwidth ptap_missing 0 bend_illegal
        "N-diff distance to P-tap must be < 15.0um (LU.2)"
 cifmaxwidth ntap_missing 0 bend_illegal
        "P-diff distance to N-tap must be < 15.0um (LU.3)"
```

All four are about **well taps**, the connections that hold the body of your transistors at a known voltage. They're left out of the default set because they are expensive to compute, and because on a single cell they usually flag a problem that isn't really there: standard cells leave their taps out on purpose. The foundry's own inverter fails three of them (see Anatomy of a standard cell).

The fourth, `licon.16`, never fires on the pinned image, not even on a tap with no contact. The layer it checks, `tap_missing_licon`, is every tap minus a layer called `tap_with_licon`, and the tech file builds `tap_with_licon` by growing outwards from every tap, contacted or not, so the subtraction always leaves nothing. A tap with no contact still gets caught, but by the rules it leaves unsatisfied: Lab 02's `drc_05` has an n-tap with no contact, and the full rule set reports `nwell.4` and `LU.3` for it.

Why a missing tap is dangerous

A missing tap is hard to catch because nothing you normally look at shows it:

- The layout looks complete.
- `drc check`, with the default rule set, reports **0**.
- The cell extracts.
- It might even simulate, depending on what the floating body does.

Here is the block you build in the capstone, a row of four standard cells called `datapath`, checked both ways before its tap cells go in:

```
      DRC, default rule set  0
      DRC, full rule set     32
```

The default check reported none of the thirty-two violations in this block. That matters because on real silicon, missing taps cause **latch-up**. Every CMOS process contains parasitic bipolar transistors between the well, the substrate and your diffusions, and the taps are what hold the well and substrate firmly at their supplies. If nothing holds them there, a transient can turn those transistors on, and once they're on they form a self-sustaining short

from supply to ground. The chip stops working until you cut the power, and sometimes it still doesn't work after that.

The rule of thumb

> **Before you believe a DRC count of zero, run it again with `drc style drc(full)`.**

Every lab in this course that runs DRC runs both rule sets and prints both counts side by side, so inside a lab you never have to remember to do this. When you work in Magic by hand, you have to switch to the full set yourself, with this sequence:

```
drc style drc(full)
drc check
drc catchup
drc count total
```

To switch back to the default rule set, run `drc style drc(fast)`.

The same fault in the netlist

A missing tap also shows up outside DRC. Two independent tools notice it, and neither of them uses the word "tap". If you extract a block with no taps and look at the netlist, you see this:

```
.subckt datapath CLK D A B Y VGND VPWR
Xbuf0 buf0/A VGND VSUBS dff0/VPB VPWR Y sky130_fd_sc_hd__buf_1
```

The bulk terminals came out as **`VSUBS`** and **`dff0/VPB`**. Those are invented node names, which appear because nothing connects those wells to a supply. Netgen then reports a mismatch, because the schematic you compare against has the same terminals on VGND and VPWR.

With the taps in place, the same line becomes:

```
Xbuf0 buf0/A VGND VGND VPWR VPWR Y sky130_fd_sc_hd__buf_1
```

The invented names and the Netgen mismatch are two symptoms of the same missing taps, and neither of them is a DRC error in the rule set you were running. Lab 02's fifth cell has this fault at the scale of a single inverter, so you meet it first in a design small enough to see all of it.

Checkers in general

What this page shows about taps holds for any checker:

> **When a checker passes, it has told you that none of the rules it ran fired, which is not the same as telling you the design is right.**

The digital track meets the same problem from a different direction. In DD103 an entire chip hardens successfully in the *wrong PDK*, behind three green ticks and a real `.gds`, because nothing in the flow checks that you meant that PDK. In both tracks the habit that protects you is the same: know which checks you're running, and ask what they aren't looking at.