# The analog layout kit

This page answers: *what does Tiny Tapeout give you to start an analog layout, and what has a layout to contain before precheck will accept it?*

A digital Tiny Tapeout project gets its layout from LibreLane, which places the pins where Tiny Tapeout wants them because Tiny Tapeout configured it to. An analog project's layout is yours, pins included, so Tiny Tapeout publishes the pieces you need to start one in the right shape, in its support tools under `tech/sky130A/def/analog/`. In this course we call those pieces the kit. You'll use them in Lab 01, Part 5, and change their settings in Lab 03.

## The DEF templates

A DEF file describes a layout's outline and its pins. The kit has one per tile size, `tt_analog_1x2.def`, `tt_analog_2x2.def` and so on up to `6x2`, each with a `_3v3` variant for projects that use the second analog supply. The analog template's `info.yaml` says "Valid values for analog projects: 1x2, 2x2", and so do Tiny Tapeout's analog specs; the 3x2 to 6x2 templates were added to the kit later, on Apr 18, 2026.

The 1x2 template, the one this course uses, describes a tile 161.00 µm wide and 225.76 µm tall, with 51 signal pins, all on metal 4 and all on the tile's edge:

- **On the top edge**, the digital pins, each 0.30 µm wide and 1 µm tall, 2.76 µm apart, from right to left: `ena`, `clk` and `rst_n`, then `ui_in[0..7]`, `uio_in[0..7]`, `uo_out[0..7]`, `uio_out[0..7]` and `uio_oe[0..7]`. Their centres run from 146.74 µm (`ena`) down to 30.82 µm (`uio_oe[7]`).
- **On the bottom edge**, the eight analog pins `ua[0..7]`, each 0.90 µm wide and 1 µm tall, 19.32 µm apart, `ua[0]` furthest right with its centre at 152.26 µm. Tiny Tapeout's specs say "Even though there are 8 pins in the templates, you can only use the first 6", and "You must use the pins in order, starting from 0."

The DEF also names the two power nets, `VDPWR` (1.8 V) and `VGND`, but doesn't place them. Where they go is up to you, within the rules below.

The `_3v3` templates add a third power net, `VAPWR`, and are narrower: the 1x2 one is 145.36 µm wide, and its analog pins sit at different x positions. So setting `uses_vapwr: true` in `info.yaml` changes a lot at once, because precheck then compares your layout with the `_3v3` template; you'll see this in Lab 02's breakage 5.

## magic_init_project.tcl

The kit's other file is a Magic script, `magic_init_project.tcl`, by Uri Shaked of Tiny Tapeout. It makes the starting point of a layout from a DEF template, and its comments ask you to edit only its settings block:

```
set TOP_LEVEL_CELL     tt_um_analog_example
set TEMPLATE_FILE      tt_analog_1x2.def
set POWER_STRIPE_WIDTH 2um                ;# The minimum width is 1.2um

# Power stripes: NET name, x position. You can add additional power stripes for each net,
as needed.
set POWER_STRIPES {
    VDPWR 1um
    VGND  4um
}
```

Then it does four things:

1. `def read` reads the template, which creates a cell with the tile's outline and every pin, and `cellname rename` gives the cell your top module's name.

2. For each entry in `POWER_STRIPES`, it draws a vertical stripe of metal 4 from y = 5 µm to y = 220.76 µm, `POWER_STRIPE_WIDTH` wide, labels it with the net's name, and makes the label a port with `port use power` (or `ground` for VGND).
3. It saves the cell as `<TOP_LEVEL_CELL>.mag`, Magic's own format.
4. It writes `gds/<TOP_LEVEL_CELL>.gds`, and `lef/<TOP_LEVEL_CELL>.lef` with `lef write -hide -pinonly`, which describes only the outline and the pins and hides the rest.

Those two paths are exactly the ones the template's `gds` workflow reads, so the script's output could go straight into a repository. On its own, though, it isn't a passing layout: the template's `info.yaml` has `analog_pins: 2`, and precheck's analog pin check fails unless `ua[0]` and `ua[1]` each touch some metal.

`TOP_LEVEL_CELL` has to be the same as `top_module` in `info.yaml`. The workflow finds the files by `top_module`, and precheck then checks that the cell inside the GDS has that name; a mismatch is Lab 02's breakage 3.

## Tiny Tapeout's rules for an analog layout

Tiny Tapeout's analog specs set the rules a layout has to follow, beyond the process's own design rules. These are the ones this course's layouts touch, in the specs' words where possible:

| Rule | What it means for your layout | Checked by |
|---|---|---|
| Power pins are "vertical stripes on met4" that "Start within the bottom 10 µm of the module and extend at least to the top 10 µm", with a "Minimum width of 1.2 um" | The kit's stripes run from 5 to 220.76 µm in a 225.76 µm tile, 2 µm wide, so they pass wherever you put them across the tile | precheck's pin check, which measures each power port's layer, width and distance from the top and bottom edges |
| "You are not allowed to use the metal 5 layer in your design." | Nothing on metal 5, even where it's legal SKY130 | precheck's forbidden layer check |
| "You must use the pins in order, starting from 0." | With `analog_pins: 2`, you use `ua[0]` and `ua[1]`, and nothing touches `ua[2]` or above | precheck's analog pin check |
| "Connect any unused uo_out, uio_out and uio_oe pins to GND" | 24 short wires to VGND | nothing automatic; LVS against a reference that includes them |
| "If you're using Magic - do not flatten the .mag file" | Keep cells as cells; the course's builds place them as instances | — |
| "Make sure the version of Magic matches the PDK." | Use precheck's Magic with precheck's PDK; see Your tools and the runner's tools | — |

Each analog pin's path to the outside of the chip also has limits ("Resistance: < 500 ohm", "Capacitance: < 5 pf", "Max current: 4 mA"), which matter for a real analog circuit and not for a practice inverter.

The template's `src/project.v` belongs with the layout too. It declares a module with your top module's name and every pin, `VGND` and `VDPWR` included, and nothing inside. The `gds` job copies it into `tt_submission` beside the GDS as `<top_module>.v`, and precheck checks that it parses and that it declares the same power pins as the LEF.

## What the course adds

The course's layout kit, which Lab 01 copies into your repository, wraps Tiny Tapeout's two files without changing them:

- `layout/init.tcl` is `magic_init_project.tcl`, and `layout/tt_analog_1x2.def` is the 1x2 template, copied with a note of where they came from in `layout/UPSTREAM.md`. You edit only the settings block of `init.tcl`, as its comments ask.
- `layout/wires.tcl` holds everything your layout adds to the frame: two short stubs on `ua[0]` and `ua[1]` in Lab 01, and in Lab 03 an inverter and its wires. Its commands are Magic's own, with two shorthands, `p` to paint a rectangle and `place` to put a cell down, and a third, `tieoff`, that draws the 24 wires from the digital outputs to a VGND stripe.
- `make` runs Tiny Tapeout's script to make the frame, then `layout/build.tcl`, which adds `wires.tcl` to it and writes `gds/` and `lef/`. Both runs use precheck's Magic 8.3.568 and PDK, from the course tools. Running Tiny Tapeout's script unchanged and adding to its result means you can always see which part of the layout came from Tiny Tapeout and which from you.

`make precheck` runs precheck on the result, `make lvs` compares it with the circuit it should contain, and `make view` opens it in Magic on the desktop. Here's the frame that Lab 01 commits, rendered the way the `gds_render` step renders it: the tile, the pins along both edges, the two power stripes at the left edge, and the two stubs at the bottom right.

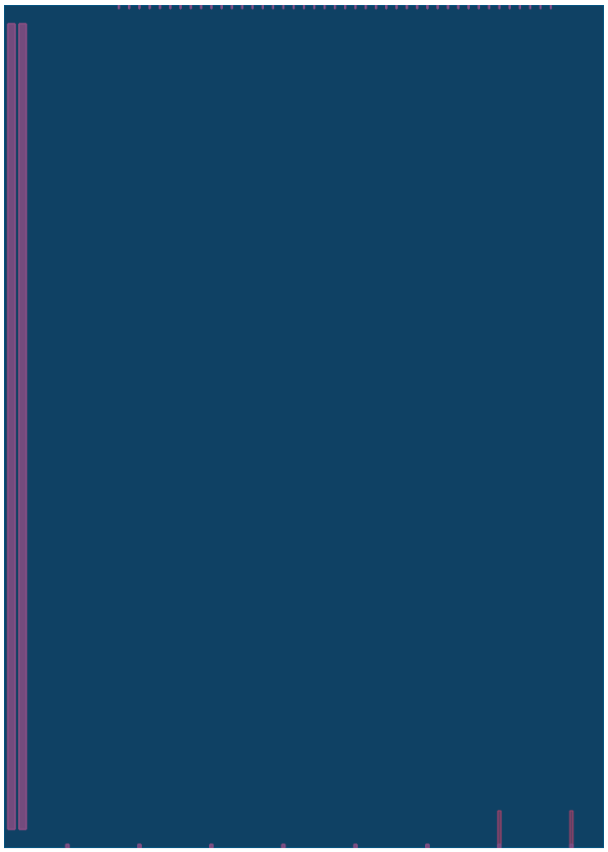


Figure 1: The Lab 01 frame: a 161 × 225.76 µm tile with pins on the top and bottom edges, two power stripes near the left edge, and short stubs on ua[0] and ua[1].

## Check yourself

1. You set `TOP_LEVEL_CELL` to `tt_um_alice_ic102` and `top_module` to `tt_um_alice_ic102a`. Which job fails, and with what?

   *Answer: A1, in Answers at the end.*

2. Could you put both power stripes at x = 150 µm and 154 µm?

*Answer: A2, in Answers at the end.*

## Answers

### A1. Check yourself

The `gds` job, at the copy step, because it looks for `gds/tt_um_alice_ic102a.gds` and Magic wrote `gds/tt_um_alice_ic102.gds`: `cp: cannot stat 'gds/tt_um_alice_ic102a.gds': No such file or directory`. (The course's `make` stops before that, with a message that the two names differ.)

### A2. Check yourself

Not without trouble, because of the one at 150 µm. Each stripe is 2 µm wide and runs from y = 5 µm up, and `ua[0]` is at x = 151.81 to 152.71 µm with its wire running up from the bottom edge, so a stripe from 150 to 152 µm would sit on top of that wire, both on metal 4. The rules allow a stripe anywhere, so where you can put one depends on your own wiring.