LAYOUT FOR MEMS

MEMS layout in EDAlogy AI Studio: curve objects, rule checking on curves, and an AI-first command architecture

Summary. EDAlogy AI Studio is a client-server layout editor: the server holds the design and does the geometry, the rendering and the design-rule check, and the client is a web page. A layout is therefore worked on from any computer with a browser, whatever its performance or platform: Linux, Windows, macOS, or a tablet such as an iPad. This overview covers what matters for MEMS layout: curve objects, DRC on curves, release holes, technology setups and parameterised cells (bundled or your own), and one command set for the menus, scripts, the AI assistant and outside agents.

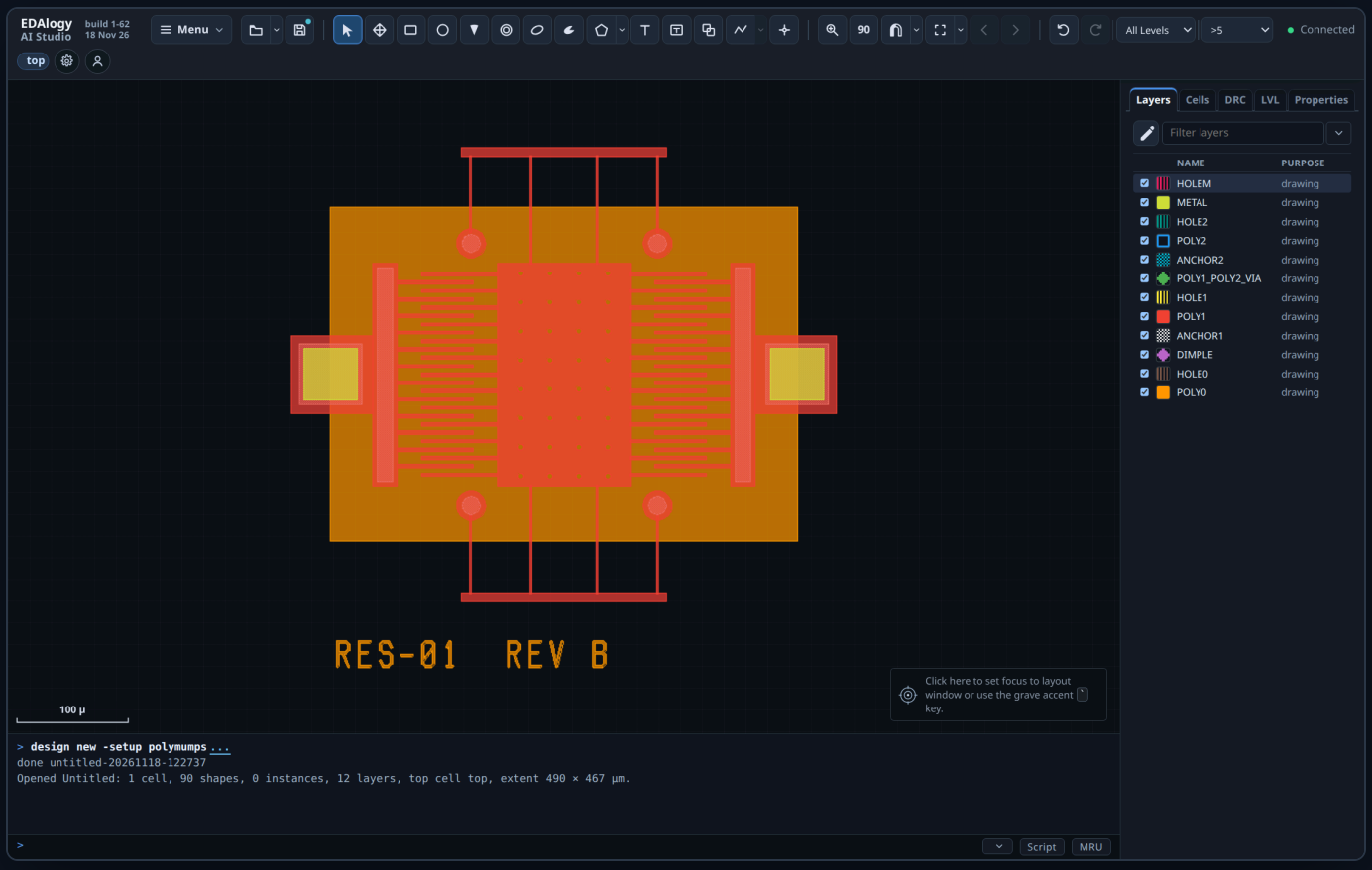

Figure 1. The studio in a browser: a comb-drive resonator in the PolyMUMPs levels, with round release holes, round anchors, a folded suspension and a die label drawn as mask text.

Rule checking on curves

On a ring gyroscope without violations, checks of the GDSII polygons give 214 markers and the studio gives none: only real errors are reviewed (section 3).

Curves as objects

Circles, rings, arcs and holes are edited by radius and angle and kept through booleans and GDSII, so no curve is redrawn (section 2).

One command set

Menus, scripts, the AI assistant and outside agents issue the same commands: what is done by hand can be scripted or asked for (section 4).

1 Technology setups and parameterised cells

A design starts from a technology setup and is built from parameterised cells (pcells). Both are bundled for the MUMPs processes, and both can be your own.

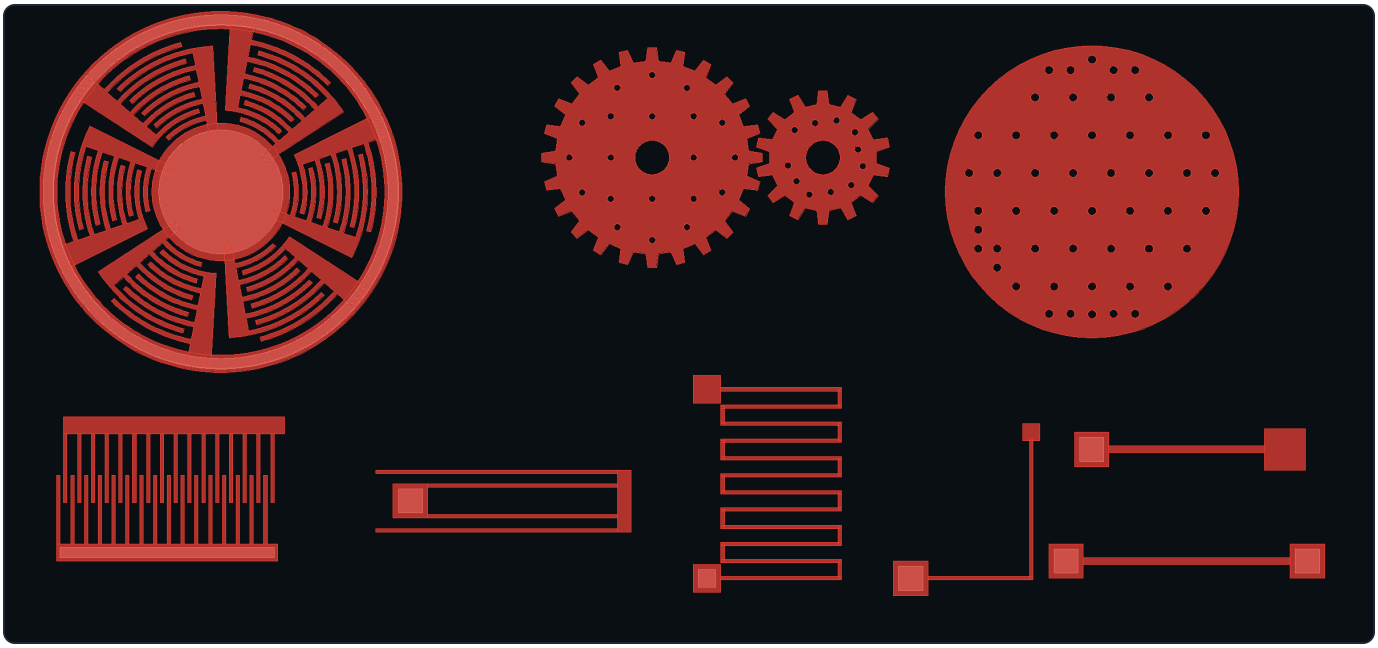

Figure 2. Cells of the bundled device library as the studio draws them: rotary comb drive, a pair of gears, perforated disc, linear comb drive, folded flexure, serpentine spring, crab leg, cantilever with paddle, bridge.

Bundled setups

A new design in PolyMUMPs, SOIMUMPs or PiezoMUMPs has the levels of the process with their GDS numbers and a width and a space rule for each level; PolyMUMPs adds its release rules. DRC works from the first shape, with nothing to configure.

Bundled devices

Ten pcells: linear and rotary comb drives; folded, serpentine and crab-leg flexures; cantilever, bridge, perforated plate, gear and anchor pad. A device is resized by its parameters, not redrawn, and its defaults pass the PolyMUMPs width and space rules as placed.

Your own technology

Any other process comes in from the files you have: KLayout layer properties (`.lyp`), a technology file with its display resources (`.tech`, `.drf`), or a setup script in Tcl or Python. Or it is set up in the studio, layer by layer and rule by rule, and exported in the same formats.

Your own pcells

A pcell is a short Tcl or Python script. The AI assistant writes one from a description of the device, which is the quick way; it can be written by hand in the built-in editor; and KLayout PCell classes and PyCell generators are ported to run in the studio's Python.

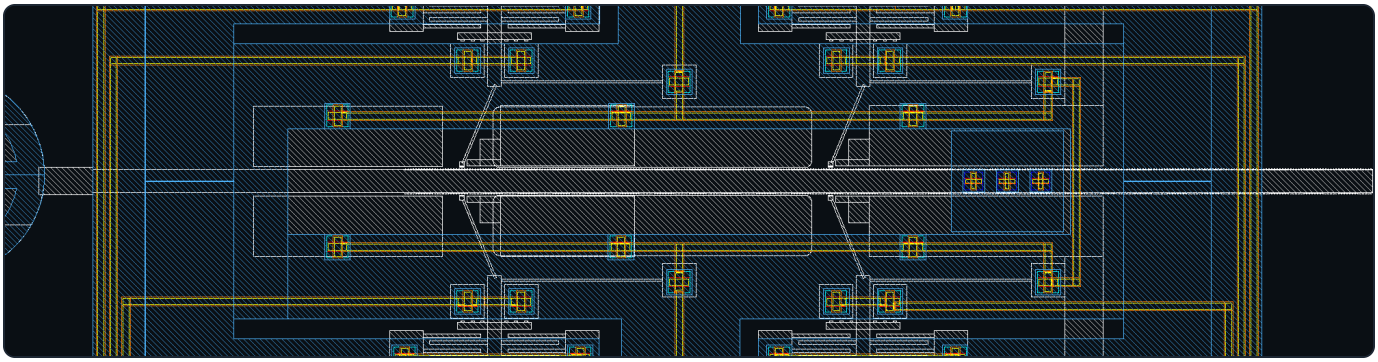

Figure 3. An existing mask read from its GDSII file: an inchworm motor from the open Aureo SOI MEMS kit (UC Berkeley).

2 Curve geometry

A curved shape is stored as an object with its parameters. It is converted to a polygon only on export to a format without curves, such as GDSII.

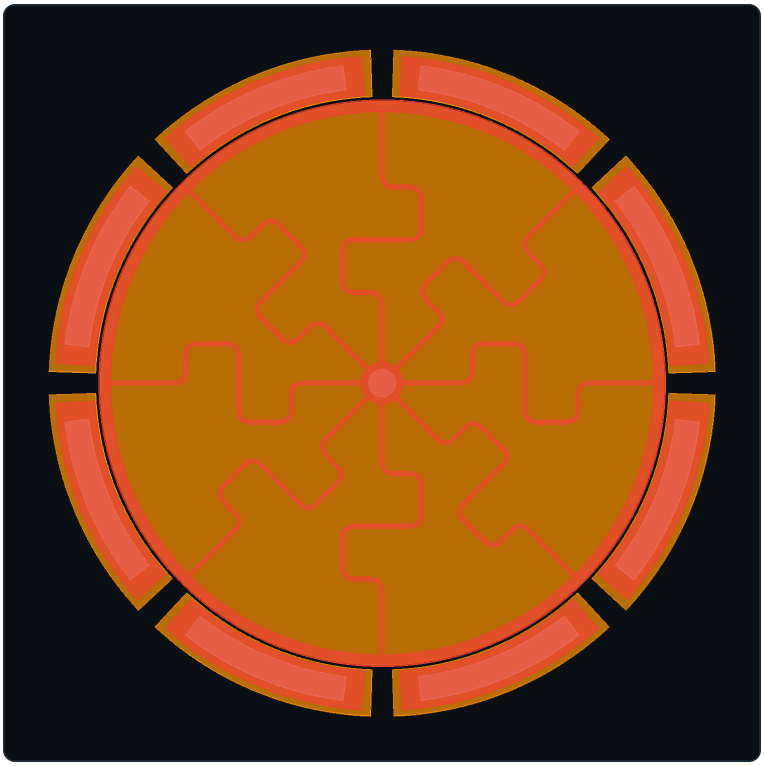

Figure 4. Test layout: a vibrating ring gyroscope. One ring, eight folded springs with rounded bends, eight electrodes, each on its own ground pad; 36 curved objects.

Editing by parameter

The ring is one object: a centre, a radius and a width. A design change is a new value, not a redrawn polygon.

GDSII round trip

On stream-out a curve becomes a polygon tagged with its object; on stream-in the object is restored. Figure 4 returned as the same 36 objects with the same radii and angles: a design leaves and re-enters the studio with its curves.

Booleans and sizing

Union, difference, grow and shrink keep arcs as arcs, and a whole ring comes back as a ring object. The result stays editable by its radius.

Bends, corners and holes

Each spring is one path with a bend radius; polygon corners take fillets or chamfers. A stress-relieving radius is one value, changed in one place.

Polygons from other tools

Circles, rings and arcs are recovered from imported polygons within a tolerance the user sets, so a legacy design becomes editable by its curves.



Figure 5. Object kinds: circle, pie, ring, ring sector and ellipse; polygon edges as arcs and splines; fillets; paths with round ends, bends and arc turns.

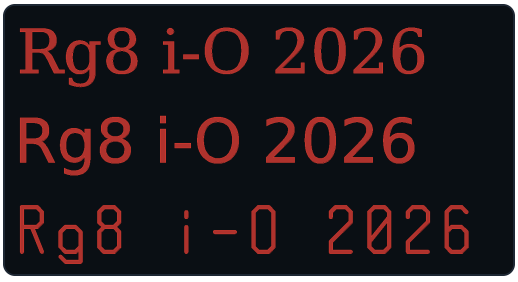

Figure 6. Text on the mask in a TrueType font or the built-in one, generated by a parameterised cell and so editable after it is placed.

3 Design-rule checking

GDSII has no curves: a ring or an arc is streamed out as a polygon of short straight segments. A width or space check of those segments marks corners where a curve meets a straight edge at just under a right angle, and breaks one real violation along a curve into many markers. The studio checks the curve itself, so the designer reviews real errors only.



Figure 7a. Width and space checks of the GDSII polygons. A yellow dot is one marker.

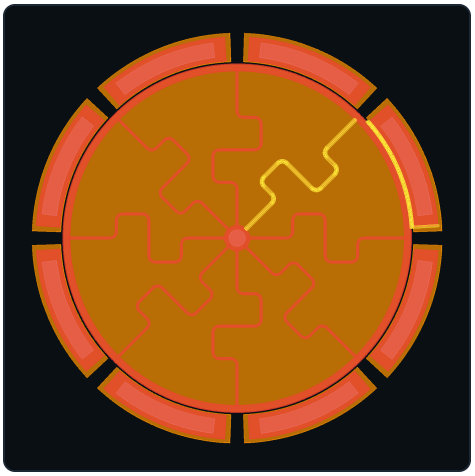

Figure 7b. The studio's check of the same layout. All markers are on the two planted errors.

LAYOUT OF FIGURE 4 (CONDITIONS: TABLE 1)	CHECK OF GDSII POLYGONS	STUDIO
As drawn, without violations	**214** markers	**0** markers
Two planted errors (Figure 7): a 1.6 µm spring, an electrode 1.5 µm from the ring	**1,096** markers	**5** markers

Release

The release rule marks every place of a layer farther than half the rule's value from its outline or a hole. `perforate` places a grid of square or round holes clear of every edge by a margin, and adds holes where the grid leaves a place uncovered, as at a round edge: a plate is perforated and proven releasable without placing holes by hand.

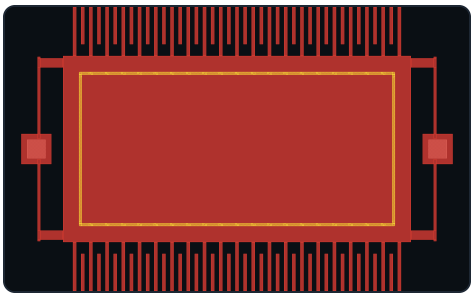

Figure 8a. A proof mass without holes. The rule outlines what the etch does not reach.

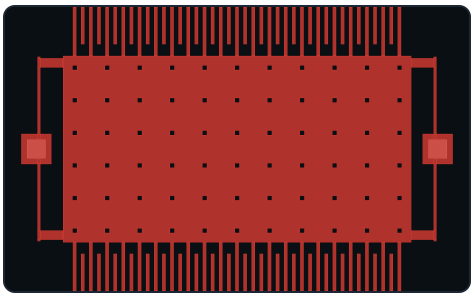

Figure 8b. After `perforate`: no markers.

Time

10,000 round holes are placed in 0.13 s and checked in 0.5 s (Table 1): perforation is redone after any change.

Checking during an edit

The pointer is held at the nearest position without a violation, so the error is never drawn.

Automatic repair

A repair is kept only if its marker clears, no rule breaks and no net changes: a fix makes no new error.

4 AI-first command architecture

The assistant is not an addition to a finished tool. Each feature was designed as a command, with its use by an agent as a design criterion, so whatever is done by hand can be asked for in plain language or scripted.

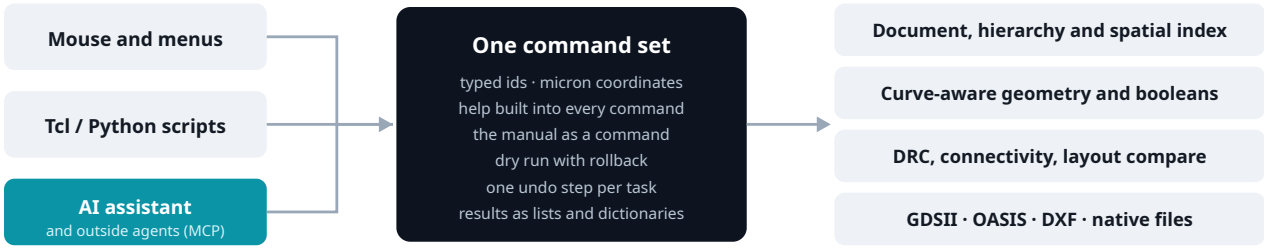

Figure 9. One command set between the three ways of driving the studio and its engine.



Figure 10. An answer of the assistant as it was given: the measured gap, the two wires by id, and a picture of the place with the gap outlined.

The session of Figure 10

Request: “Two Metal 1 wires run side by side and are very close. Tell me how wide the gap is and show me where it is.”

Execution: 13 read-only queries, then the answer; 15 s; about 0.7 cents with Claude Haiku 5.5. The design was not changed.

The same commands for user and assistant

A request such as “perforate the proof mass and check it” runs `perforate` and `drc run`. The console shows each command, so the assistant’s work can be read, repeated and kept as a script.

Dry run and undo

The assistant can try an edit on the design, read the result and roll it back. A finished task is one undo step.

Verification as commands

DRC, connectivity extraction and layout comparison are commands, so the assistant checks its result before it reports.

One manual for user and assistant

Each command and each operation is explained in the built-in manual, in a form equally useful to the user and to the AI assistant. The assistant reads it as it works; no model is retrained for a new command.

Scripting

Tcl and Python share the command set: a flow is automated in the language a team already uses.

Models and data

Any major hosted model or a local one, with a cost limit for each request. A library can be withheld from the model, so confidential cells are not disclosed.

MCP for outside agents

The studio is an MCP server: an agent the user already works with (a coding assistant, a simulation or optimisation agent, a company’s own flow) connects to it and runs the same commands as the built-in assistant. Such an agent can sweep the parameters of a device, simulate each variant in its own tool and write the chosen one back, which makes the studio one step of a larger automated flow.

```
# Listing 1. A rotary comb drive in PolyMUMPs, placed and checked
design new -setup polymumps
library attach -name mems
instance {mems ROTARY_COMB} 0 0 -params {sectors 6 fingers 5 span 30 hub_r 40}
drc run
```

5 Integration

A process, a device library, a simulator or a foundry flow connects to the studio at the points below.

Your process as a setup

A setup holds the levels, GDS numbers, width and space rules and release rules of a process. Offered in the File menu, it lets a designer start with nothing to configure.

Your devices as a library

Devices and test structures are pcells in Tcl or Python. A library is attached read-only: every designer places the same cells and none can alter them.

External tools

A simulator front end or a kit checker drives the studio by script or over MCP and reads curves by their parameters: a solver gets a radius, not a polygon to fit.

Deployment

Hosted at edalogy.com with nothing to install, or the same server inside a private network with local models, where no design data leaves the site.

Table 1. Measurements quoted in this overview

MEASUREMENT	RESULT	CONDITIONS
Markers on the ring gyroscope of Figure 4, which has no violations	**0** studio **214** GDSII polygons	PolyMUMPs minimum feature and space. GDSII polygons: the width and space checks of KLayout 0.30, default options, on the file the studio wrote.
The same with two planted errors	**5** studio **1,096** GDSII polygons	
Curved objects restored from a GDSII file	**36 of 36**	The layout of Figure 4; radii and angles as drawn.
10,000 round release holes	**0.13 s** placed **0.5 s** checked	A width and a space rule. November 2026, a workstation comparable with the ones deployed at edalogy.com.
Library devices within the PolyMUMPs width and space rules as placed	**10 of 10**	Default parameters.

Feature summary

- Circle, pie, ring, ellipse; arc and spline edges; fillets and chamfers
- Wires with end styles, joins and bend radius; any angle
- Holes of every kind; release holes, square or round; a release rule
- Booleans, grow and shrink that keep curves; verified curve recovery
- DRC on curves and holes; live DRC; verified automatic fixing
- GDSII, OASIS and DXF with curves kept; packed files; two native formats
- Hierarchy, rectangular and polar arrays, 64-bit coordinates
- Mask text in TrueType fonts; parameterised cells in Tcl and Python
- Connectivity, net tracing, SPICE export, layout versus layout
- PolyMUMPs, SOIMUMPs and PiezoMUMPs setups; a MEMS device library
- AI assistant with any major model or a local one; MCP for outside agents
- Server on Linux or Windows, or hosted; client in any browser; English and Japanese

Try it

A trial account runs the studio in your browser, with nothing to install: the examples, the MEMS device library, the process setups and the AI assistant. To ask for one, fill out the short form at **edalogy.com/trial**.

PolyMUMPs, SOIMUMPs and PiezoMUMPs are processes of Science Corporation. The Aureo kit is the work of the Pister lab at UC Berkeley, used under its BSD licence; no endorsement is implied. KLayout and the other products named belong to their owners.