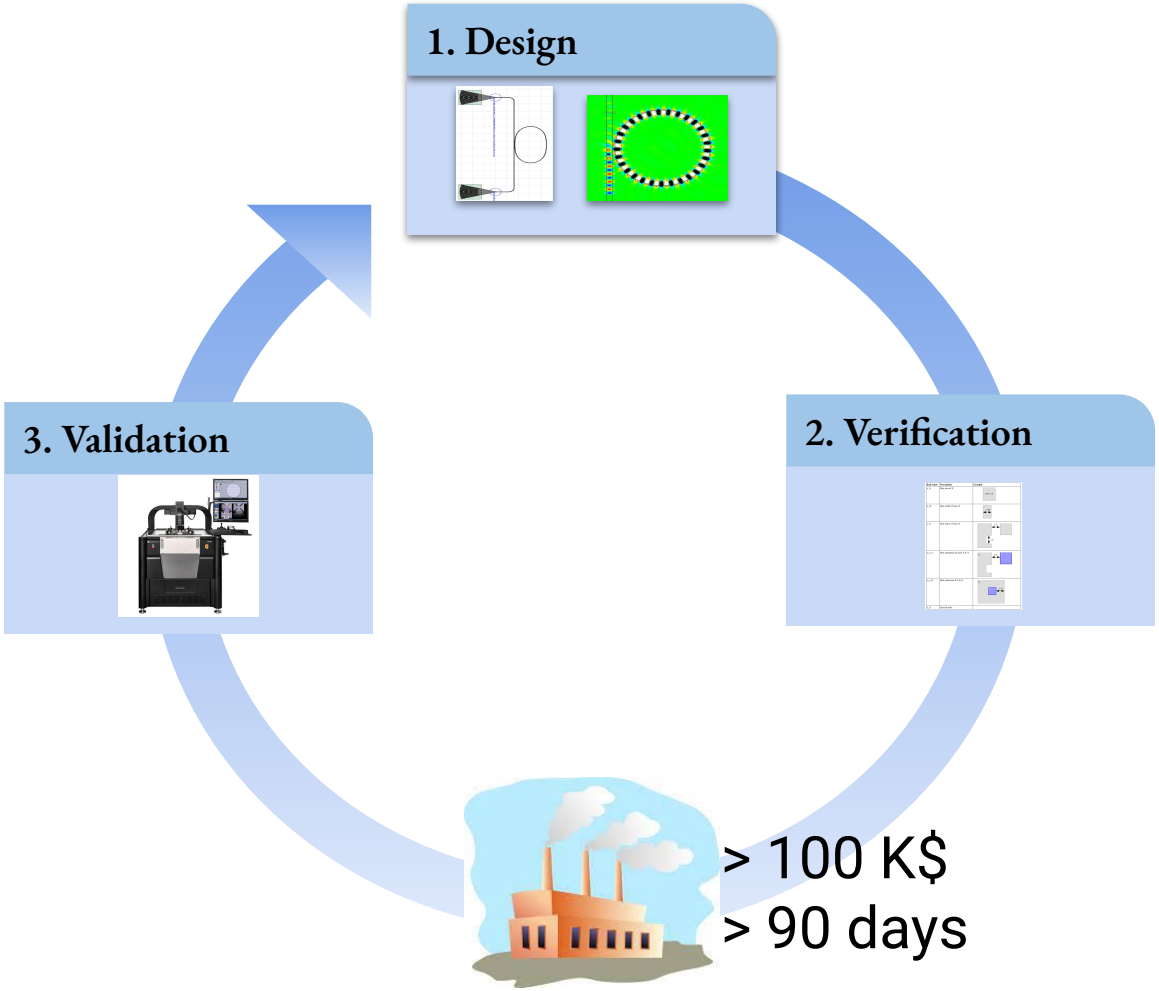
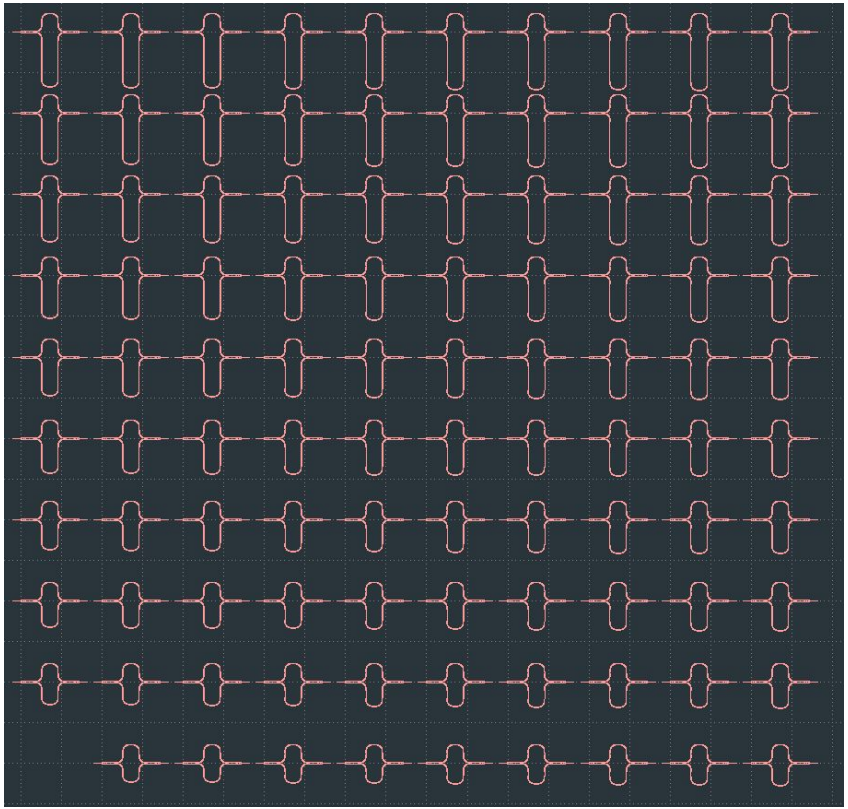
An **Open Source** flow for circuit design, verification and validation

1M+ Downloads
50+ Contributors
10+ PDKs


1. Design
2. Verification
3. Validation
> 100 K$
> 90 days

Current Layout tools are slow and hard to use



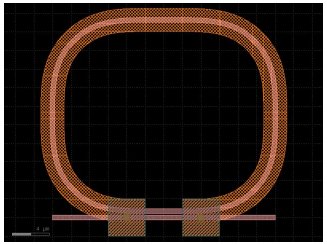
gdsfactory
100 MZI variations in
1.4 seconds

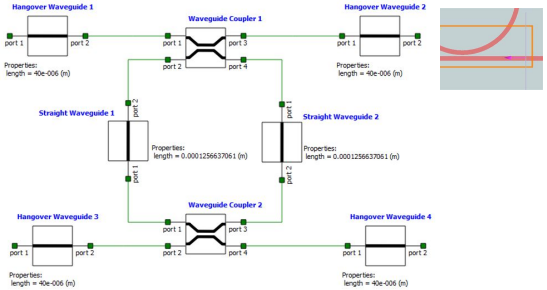
Benchmark	gdspy	gdsfactory	Gain
10k_rectangles	80.2 ms	4.87 ms	16.5
boolean-offset	187 µs	44.7 µs	4.19
bounding_box	36.7 ms	170 µs	216
flatten	465 µs	8.17 µs	56.9
read_gds	2.68 ms	94 µs	28.5

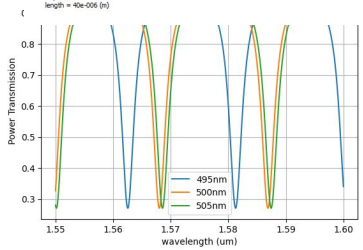
Workflow

Design

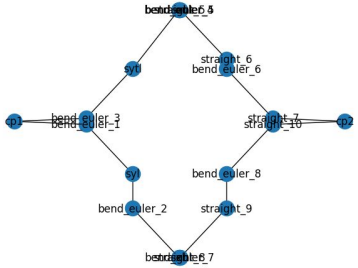
```
import gdsfactory as gf
c = gf.components.ring_single_heater(gap=0.2, radius=10, length_x=4)
```

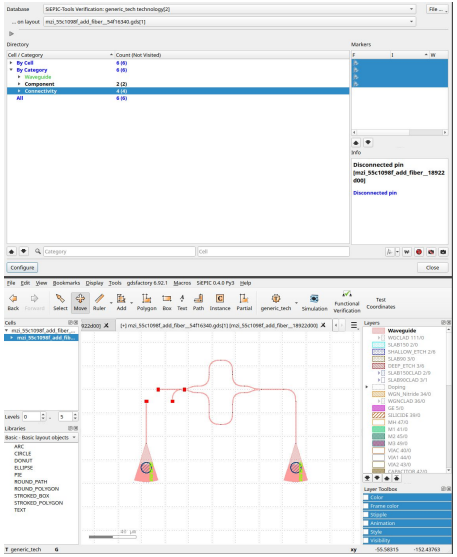




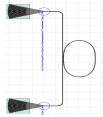


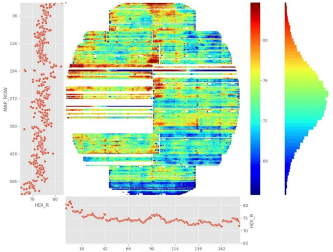
Verification





Validation





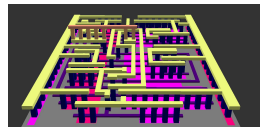
FEMWELL

Device and Circuit simulations

Ansys LUMERICAL

Tidy3D

Soux



3D view

Simulation API

Python Environment

Design Feedback

Layout + DRC API

PROBE SW Test Plan

Test Feedback

Tech File and PDK

GDS FACTORY

Wafer Level Prober

Wafer Level Die Bonder

Tower Semiconductor Where Analog and Value Meet

GlobalFoundries

CompoundTek

skywater

amf ADVANCED MICRO FOUNDRY

AIM PHOTONICS

CORNERSTONE

VTT

Fraunhofer HHI

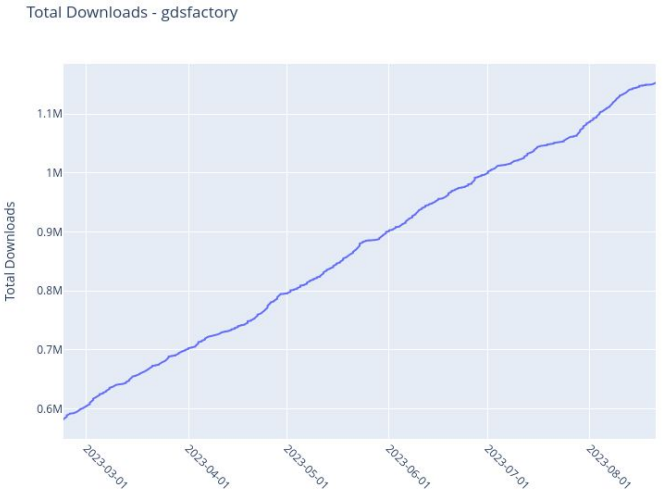
SMART PHOTONICS

ANT APPLIED NANOTOOLS

CNM Centre Nacional de Microelectrònica IMB

GDSFACTORY

- Highlights
  - 2M+ downloads
  - 60+ contributors
  - 10+ PDKs
- 100% Open source, Python based
  - Works on Linux, Windows and MacOs
- Extensible Plugins




Google


PsiQuantum


FREEDOM
PHOTONICS
A LUMINAR COMPANY


Meta


SRI International


LightIC
Technologies


UNIVERSITY OF
MICHIGAN


amf
ADVANCED
MICRO
FOUNDRY


UNIVERSITY OF
MARYLAND


Georgia
Tech


black
semiconductor


EHVA
Photonics


AIM
PHOTONICS


Lumiphase


Massachusetts
Institute of
Technology


University of
BRISTOL


PRINCETON
UNIVERSITY


Queen's
UNIVERSITY

Testimonials

"I've used **gdsfactory** since 2017 for all my chip tapeouts. I love that it is fast, easy to use, and easy to extend. It's the only tool that allows us to have an end-to-end chip design flow (design, verification and validation)."

Joaquin Matres - **Google**

"The best photonics layout tool I've used so far and it is leaps and bounds ahead of any commercial alternatives out there. Feels like gdsfactory is freeing photonics."

Hasitha Jayatilleka - **LightIC Technologies**

"I've relied on **gdsfactory** for several tapeouts over the years. It's the only tool I've found that gives me the flexibility and scalability I need for a variety of projects."

Alec Hammond - **Meta Reality Labs Research**

"I use gdsfactory for all of my photonic tape-outs. The Python interface makes it easy to version control individual photonic components as well as entire layouts, while integrating seamlessly with KLayout and most standard photonic simulation tools, both open-source and commercial.

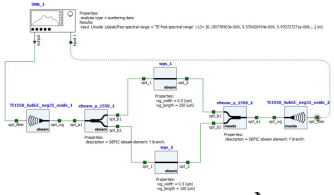
Thomas Dorch - **Freedom Photonics**

"As an academic working on large scale silicon photonics at CMOS foundries I've used gdsfactory to go from nothing to full-reticle layouts rapidly (in a few days). I particularly appreciate the full-system approach to photonics, with my layout being connected to circuit simulators which are then connected to device simulators. Moving from legacy tools such as gdspy and phidl to gdsfactory has sped up my workflow at least an order of magnitude."

Alex Sludds - **MIT**

GDSfactory PDKs

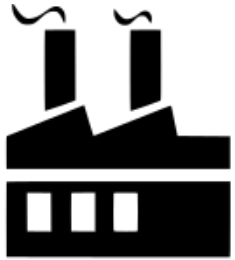
Text editor or
Schematic Editor



Python
interpreter

Klayout
GDS viewer

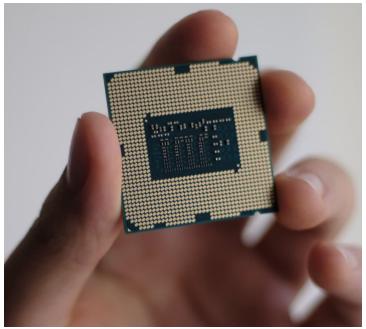
CMOS Foundry



gdsfactory python
or YAML file

GDS file

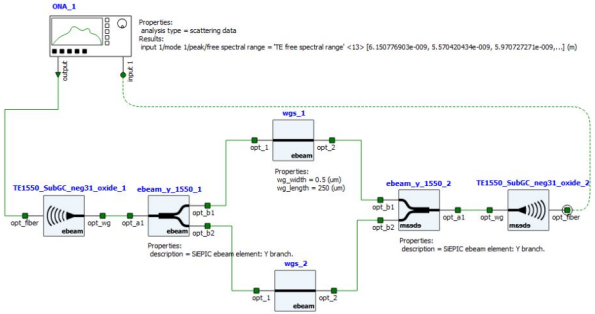
GDS, LVS and
DRC results



Physical Chip

Schematic Driven layout

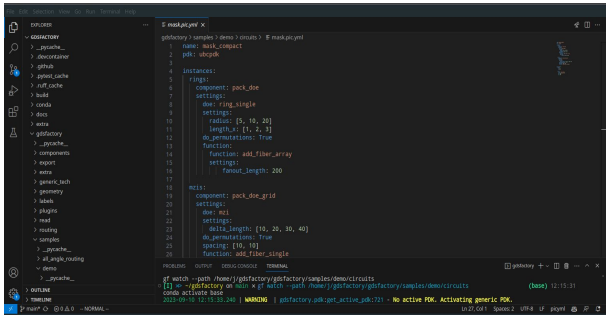
1. Design Schematic

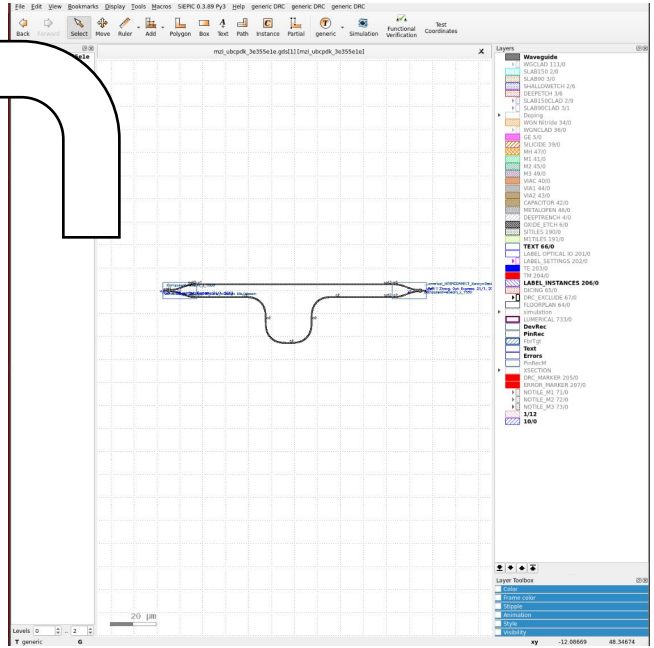


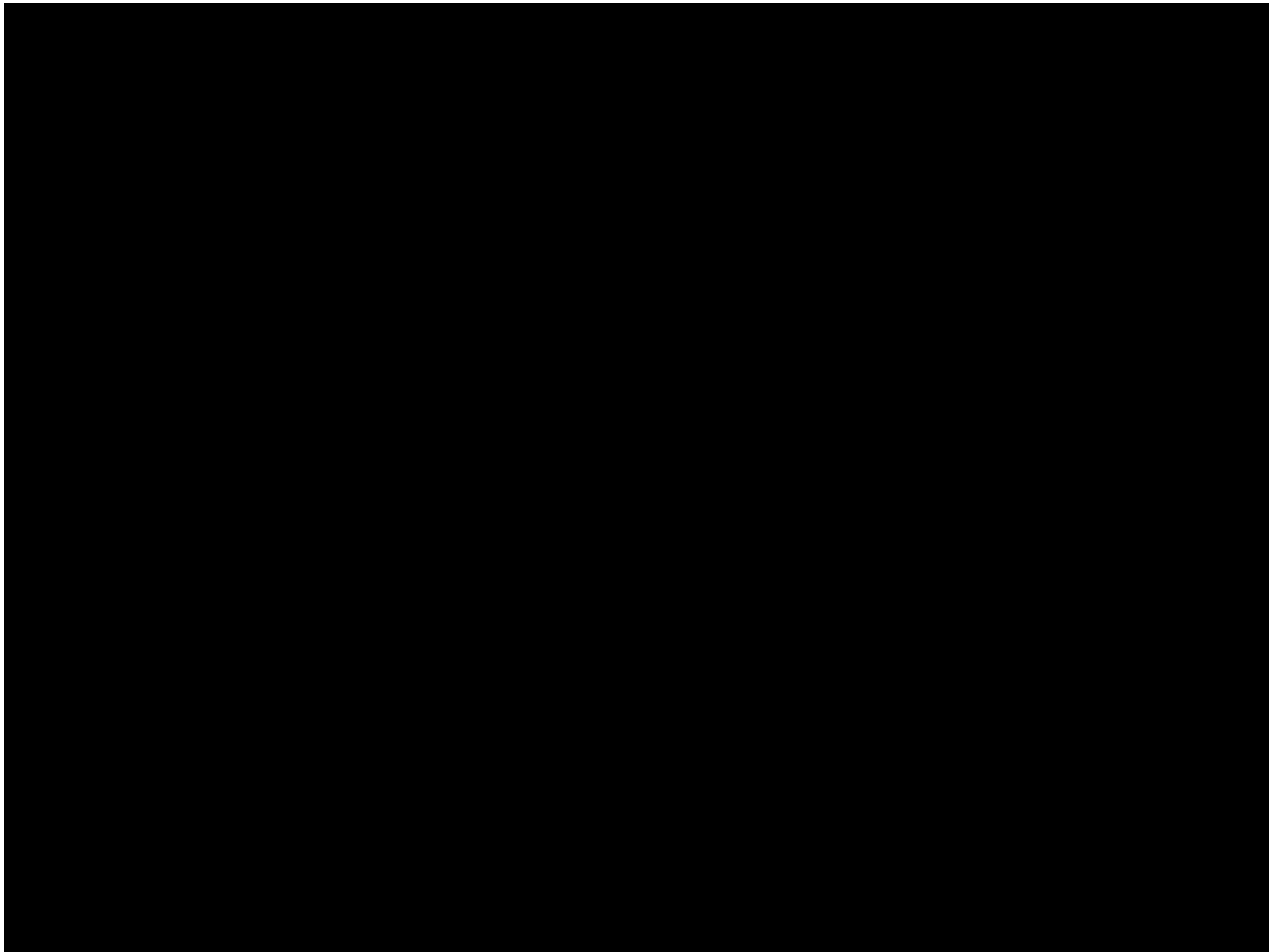
SPICE

2. Fine tune placements

YAML





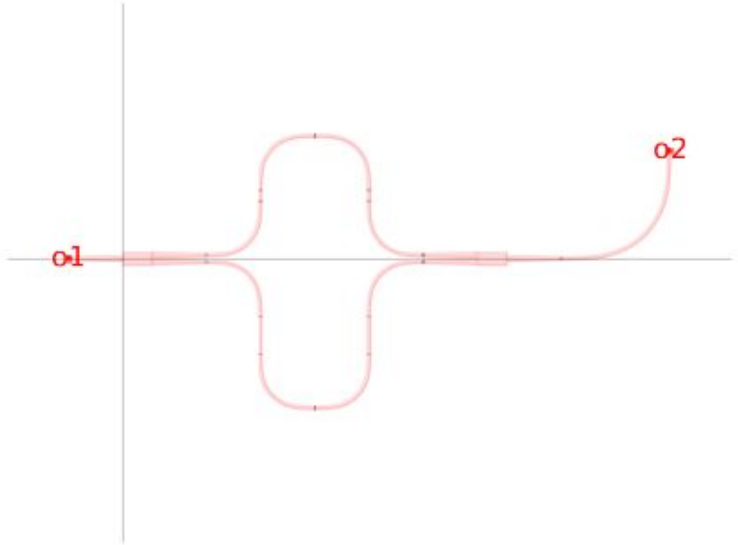


Python flow

1. run python code

2. Visualize GDS

```python
import gdsfactory as gf

@gf.cell
def mzi_with_bend(radius:float=10)->gf.Component:
      c = gf.Component()
      mzi = c.add_ref(gf.components.mzi())
      bend = c.add_ref(gf.components.bend_euler(radius=radius))
      bend.connect('o1', mzi['o2'])
      c.add_port('o1', port=mzi['o1'])
      c.add_port('o2', port=bend['o2'])
      return c


c = mzi_with_bend(radius=100)
c.show()
```

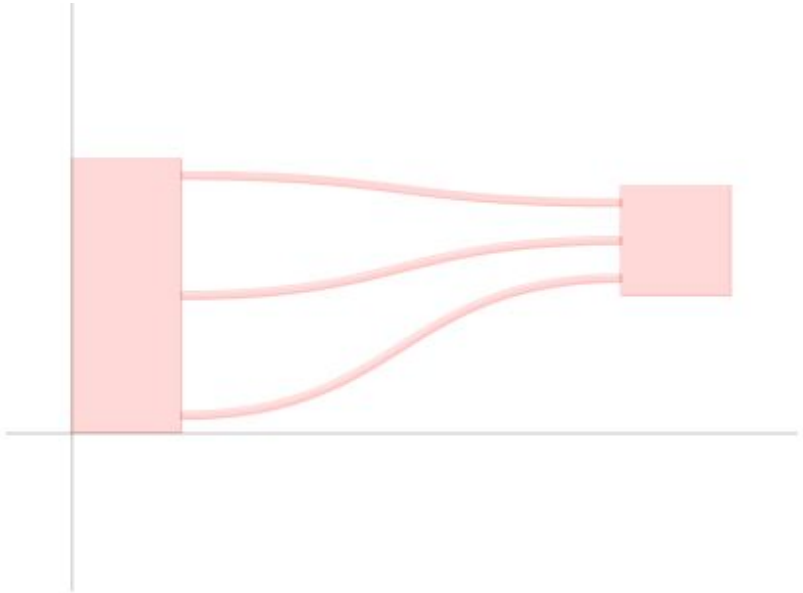


Python flow

1. run python code

```python
@gf.cell
def nxn_to_nxn() -> gf.Component:
      c = gf.Component()
      c1 = c.add_ref(gf.components.nxn(east=3, ysize=20))
      c2 = c.add_ref(gf.components.nxn(west=3))
      c2.move((40, 10))
      routes = gf.routing.get_bundle(
        c1.get_ports_list(orientation=0),
        c2.get_ports_list(orientation=180),
        with_sbend=True,
        enforce_port_ordering=False,
      )
      for route in routes:
         c.add(route.references)
      return c


c = nxn_to_nxn()
c.show()
```

2. Visualize GDS

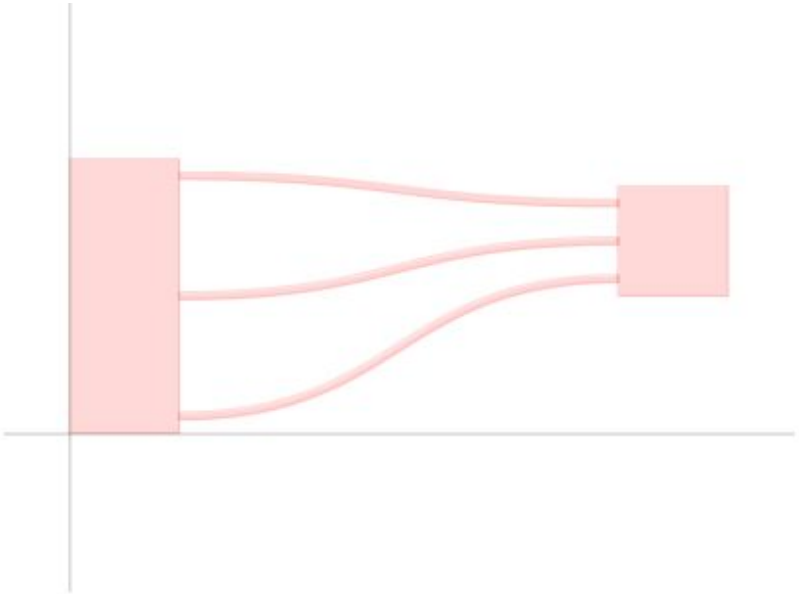


YAML flow

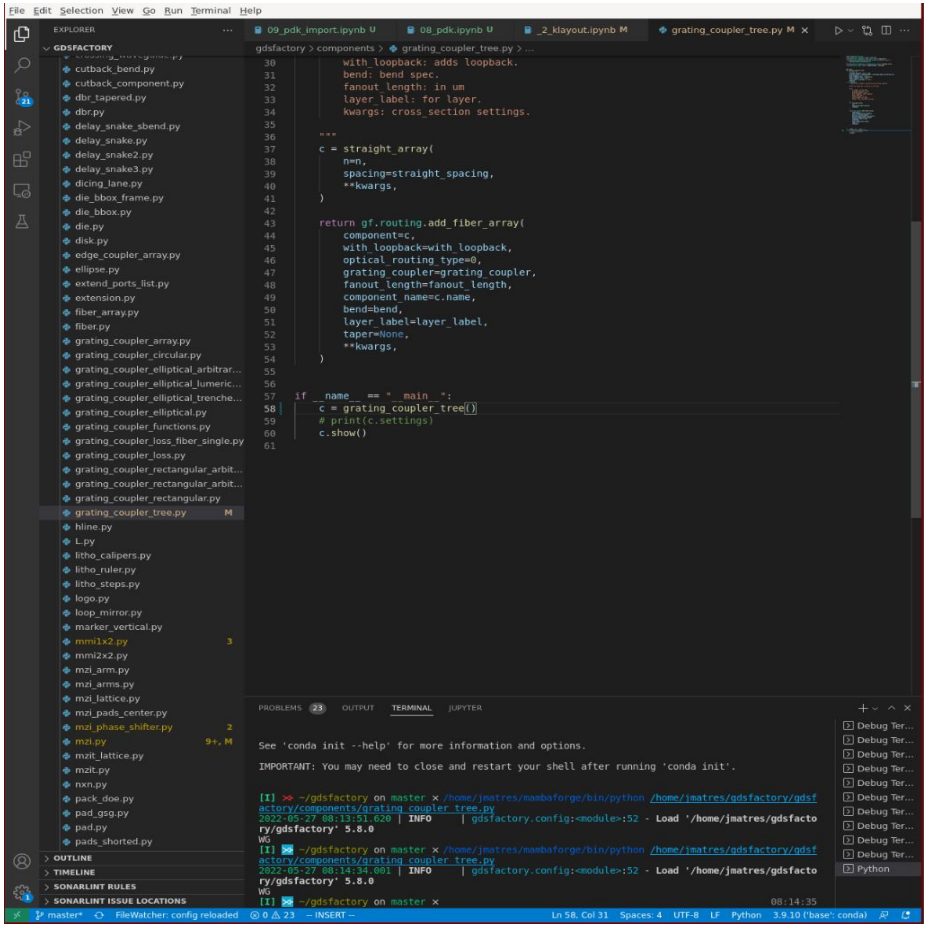
1. YAML

```yaml
name: nxn_to_nxn
instances:
  c1:
        component: nxn
        settings:
        east: 3
        ysize: 20
  c2:
        component: nxn
        settings:
        west: 3
placements:
  c2:
        x: 40
        y: 10
routes:
  optical:
    routing_strategy: get_bundle
    settings:
      with_sbend: True
    links:
      c1,o4: c2,o1
      c1,o3: c2,o2
      c1,o2: c2,o3
```

2. See GDS in klayout

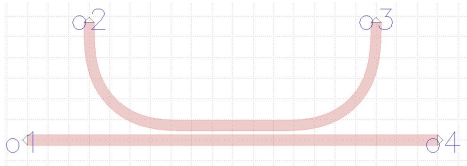


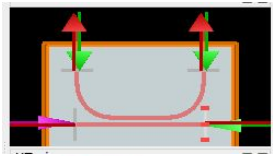
1. run python code



2. Visualize GDS or run Simulation

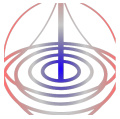
Ansys
LUMERICAL



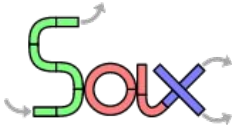


Tidy3D

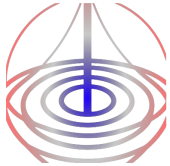
DEVSIM
TCAD software



FEMWELL



Plugins to **Open source** and **proprietary** tools



FEMWELL

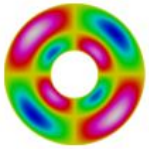
DEVSIM
TCAD software

Palace

Ansys
LUMERICAL

Soux

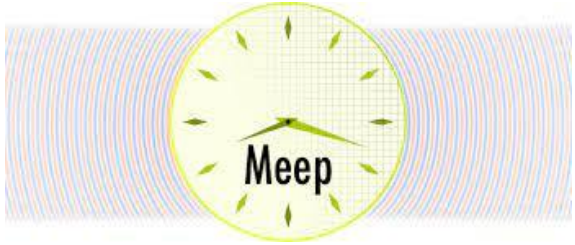
GDS
FACTORY

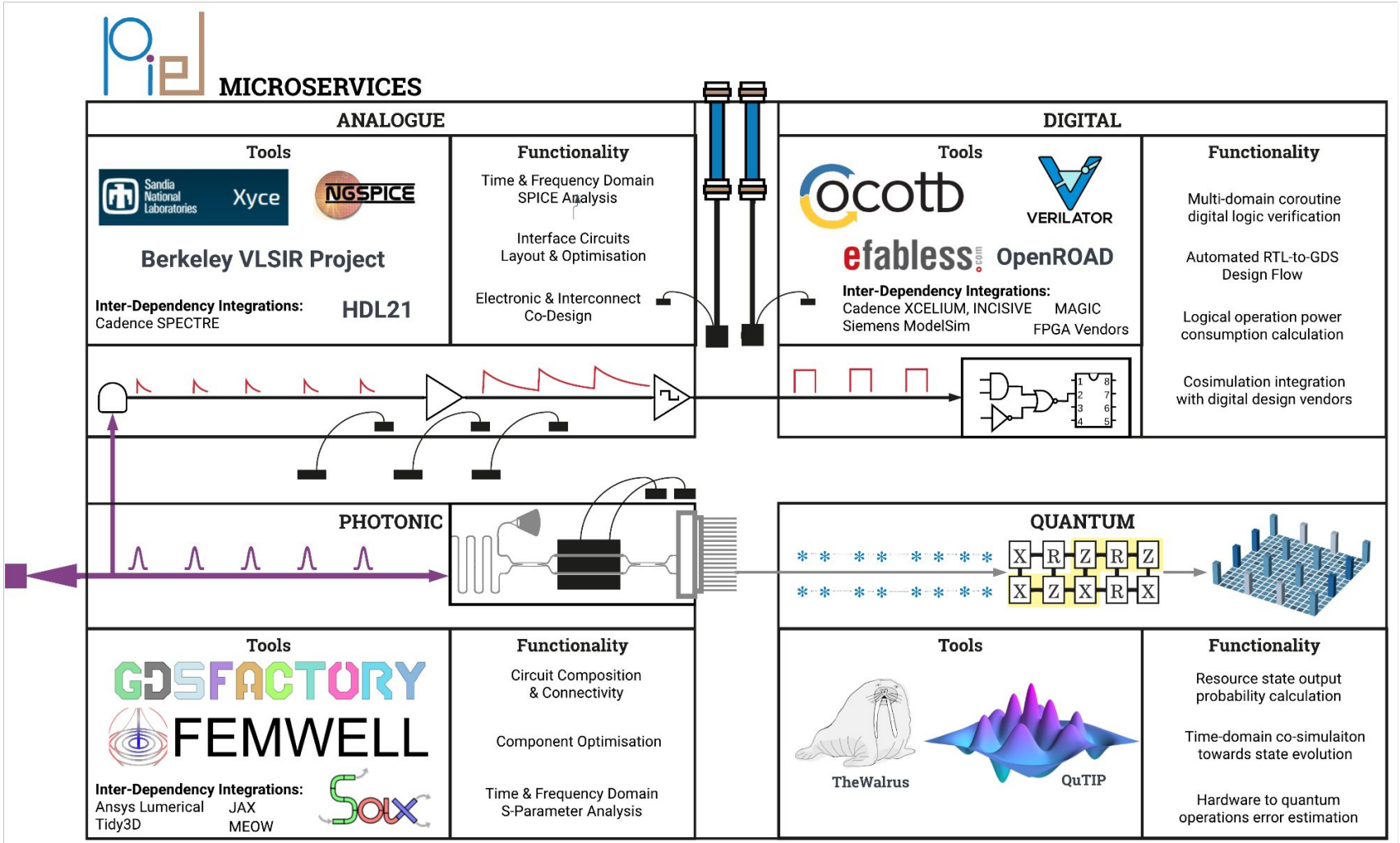


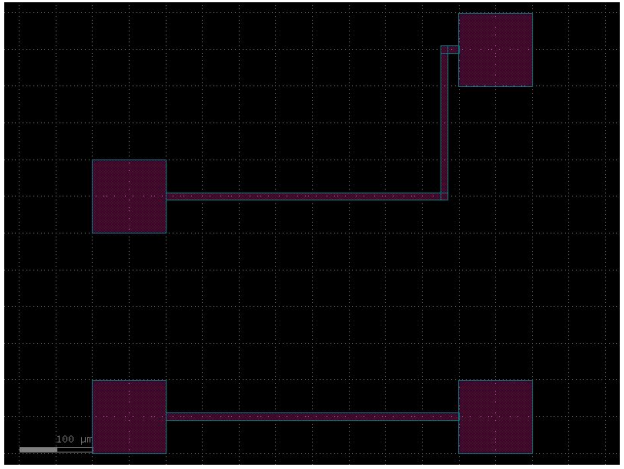
Elmer FEM
open source multiphysical simulation software

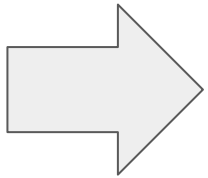
Xyce™
PARALLEL ELECTRONIC SIMULATOR

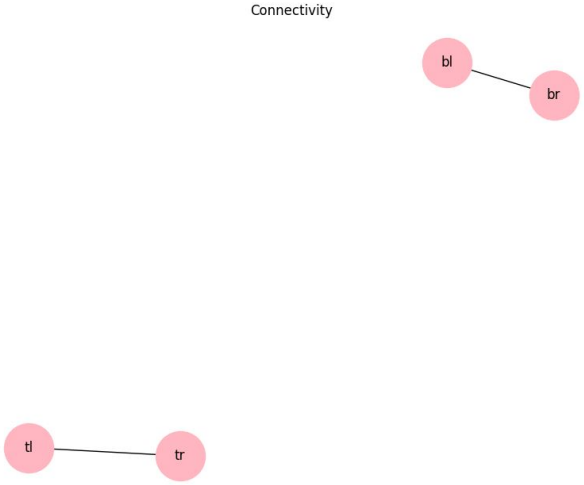
Tidy3D

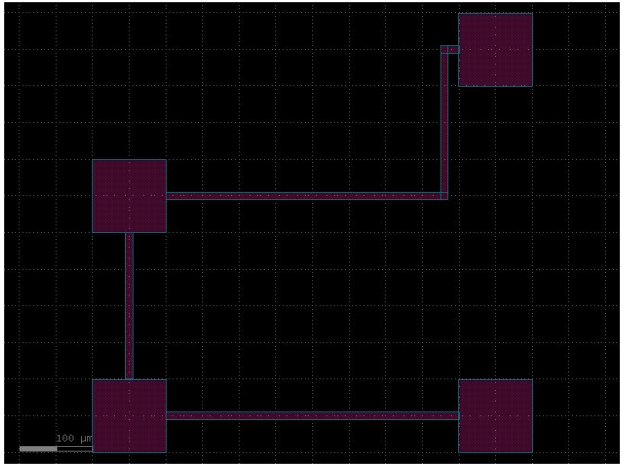


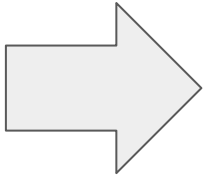

MICROSERVICES
ANALOGUE
Tools
Sandia National Laboratories
Xyce
NGSPICE
Berkeley VLSIR Project
Inter-Dependency Integrations:
Cadence SPECTRE
HDL21
Functionality
Time & Frequency Domain SPICE Analysis
Interface Circuits Layout & Optimisation
Electronic & Interconnect Co-Design
DIGITAL
Tools
ocotb
VERILATOR
efabless.com
OpenROAD
Inter-Dependency Integrations:
Cadence XCELIUM, INCISIVE
Siemens ModelSim
MAGIC
FPGA Vendors
Functionality
Multi-domain coroutine digital logic verification
Automated RTL-to-GDS Design Flow
Logical operation power consumption calculation
Cosimulation integration with digital design vendors
PHOTONIC
QUANTUM
Tools
GDSFACTORY
FEMWELL
Inter-Dependency Integrations:
Ansys Lumerical
Tidy3D
JAX
MEOW
SAX
Functionality
Circuit Composition & Connectivity
Component Optimisation
Time & Frequency Domain S-Parameter Analysis
Tools
TheWalrus
QuTIP
Functionality
Resource state output probability calculation
Time-domain co-simulaiton towards state evolution
Hardware to quantum operations error estimation

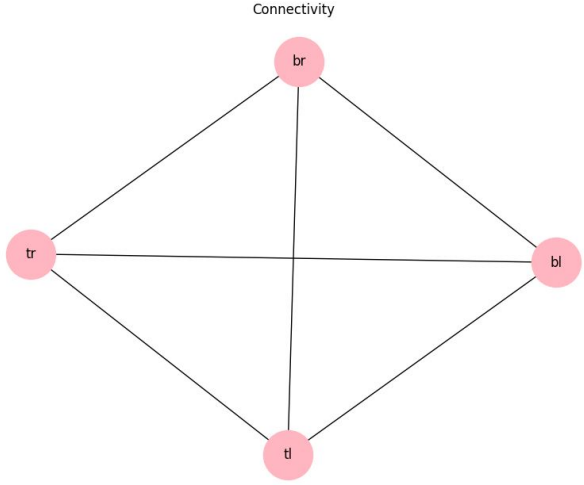

100 µm



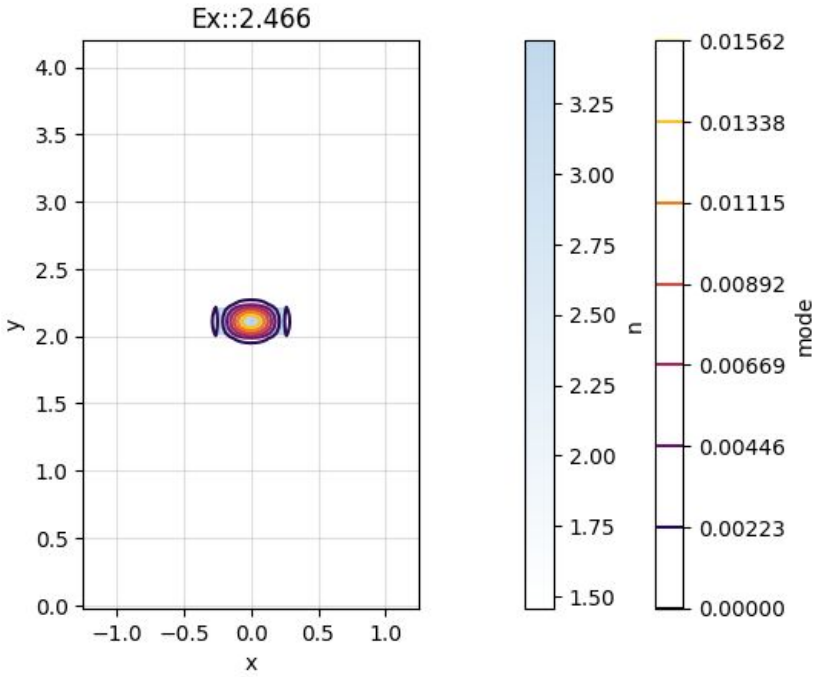

Connectivity
bl
br
tl
tr

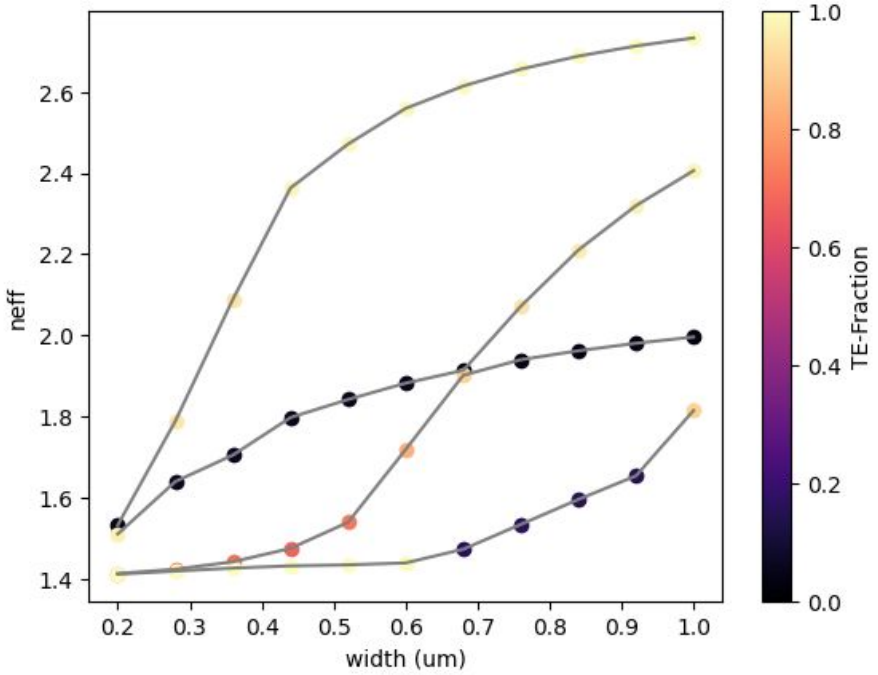

100 µm



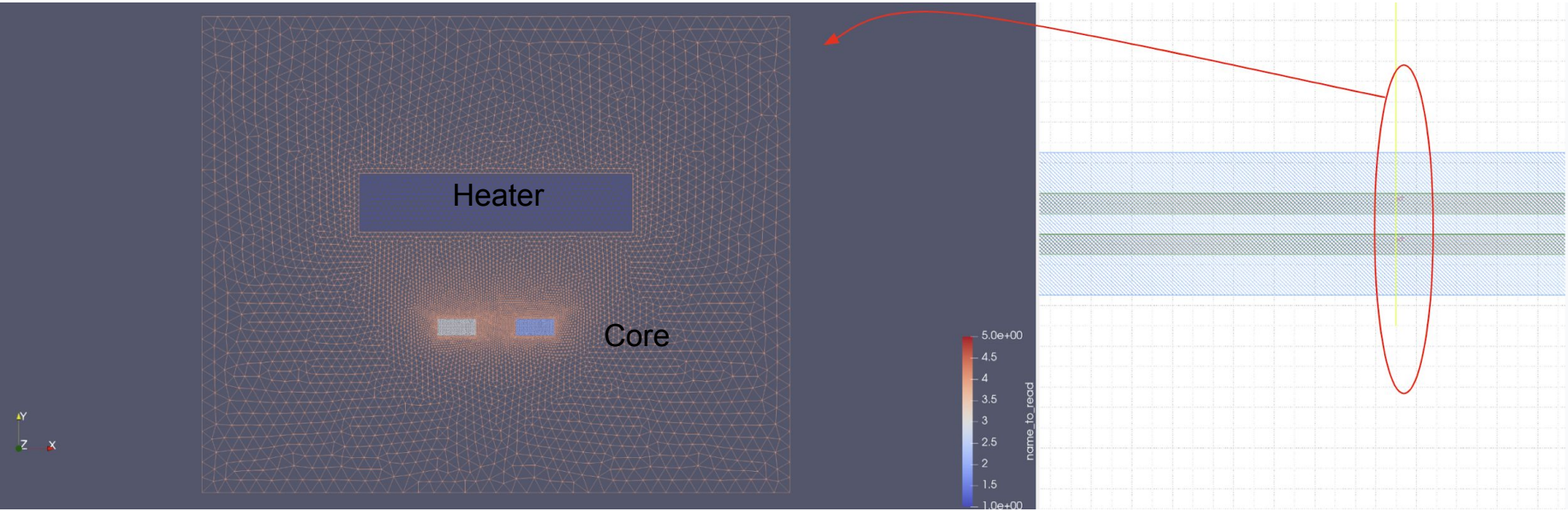

Connectivity
br
tr
bl
tl

Mode solver

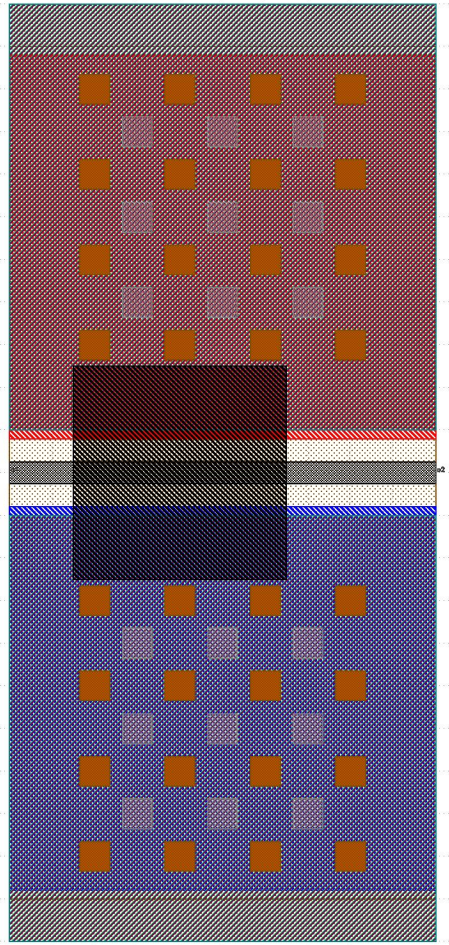




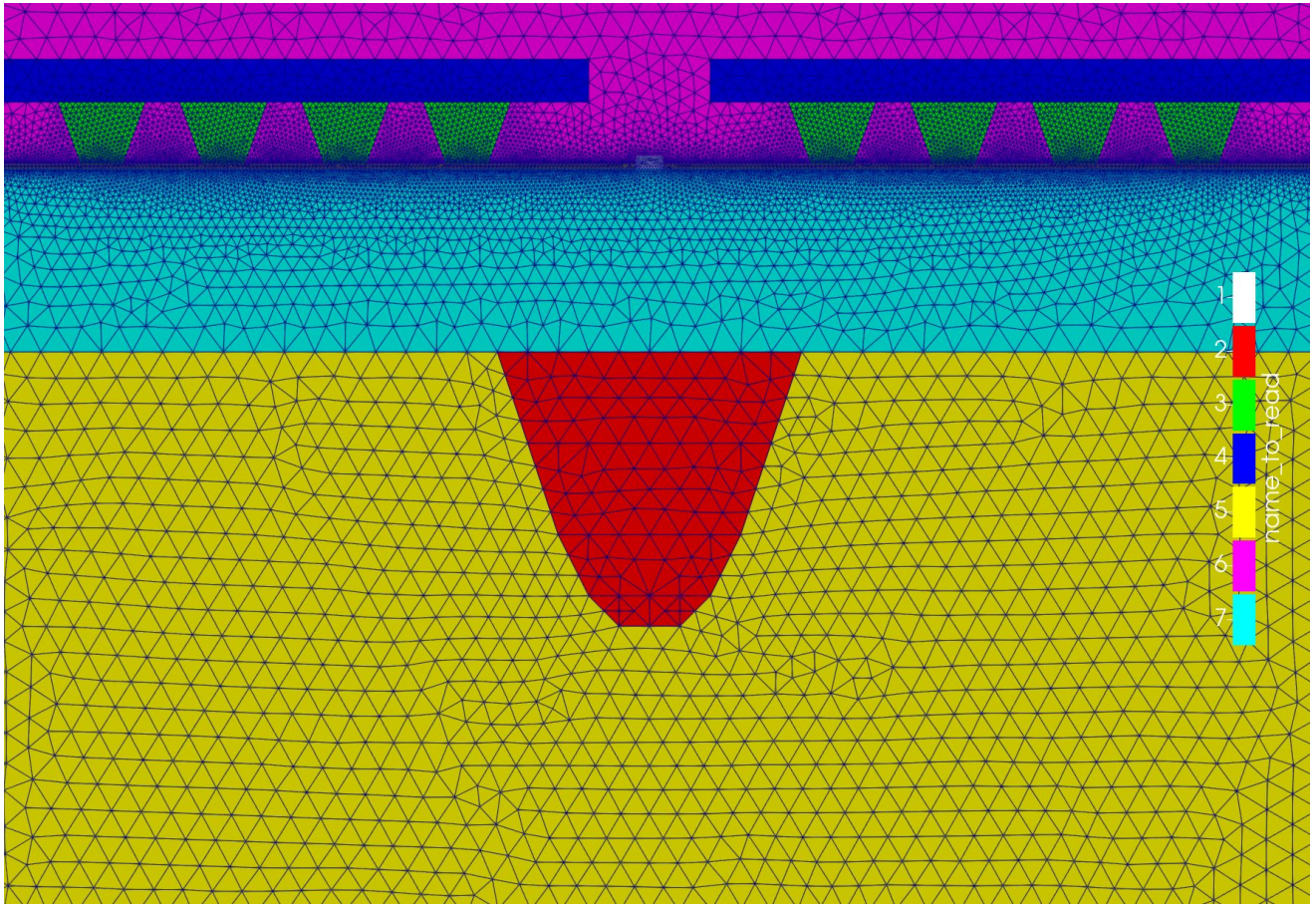
FEM mesh

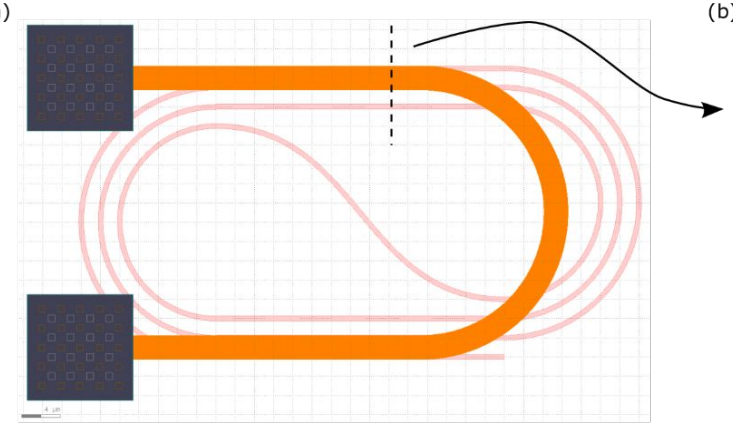


Top view

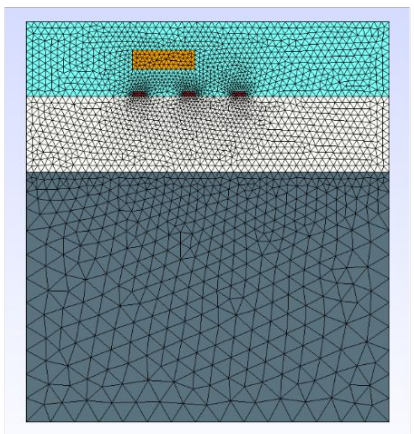


Side view

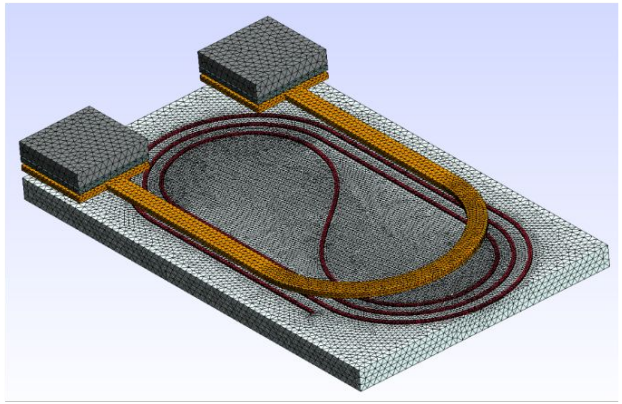




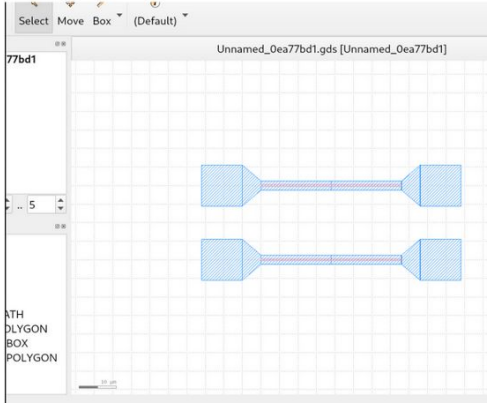
(b)

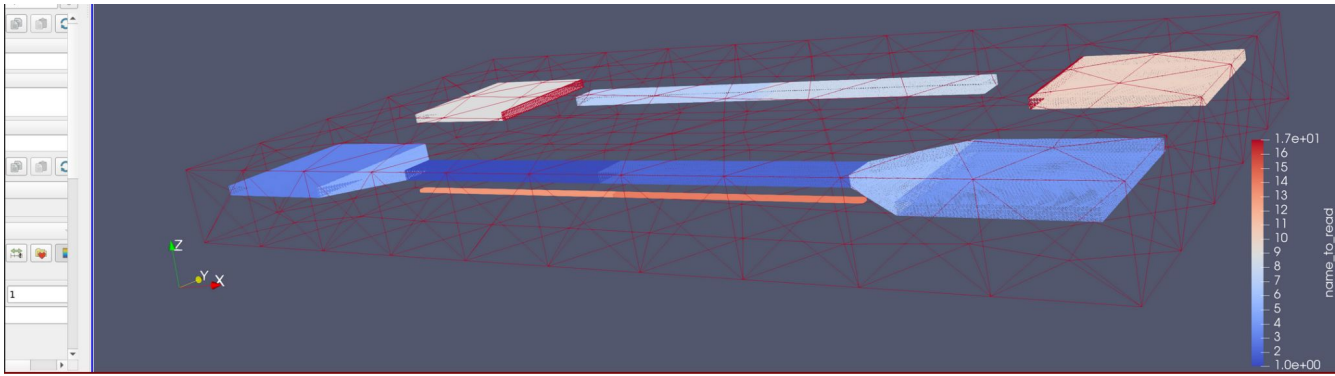


(c)



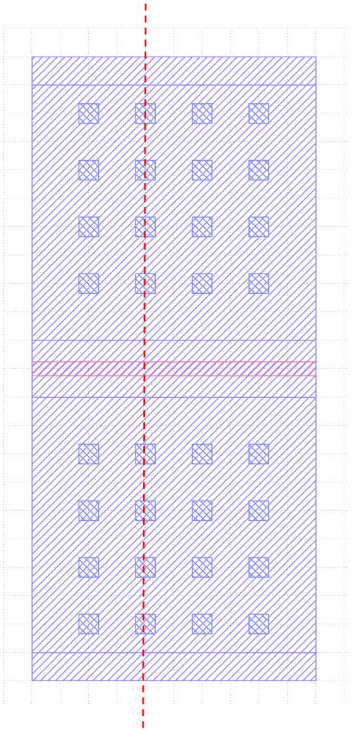
FEM mesh

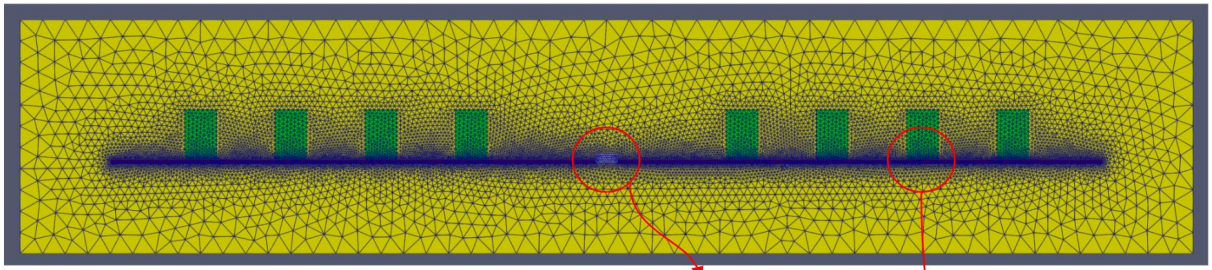

Select
Move
Box
(Default)
Unnamed_0ea77bd1.gds [Unnamed_0ea77bd1]
77bd1
5
ATH
OLYGON
BOX
POLYGON

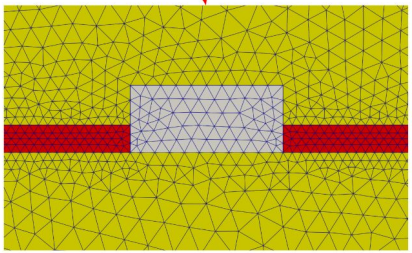

1
Z
Y
X
1.7e+01
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1.0e+00
name_to_read

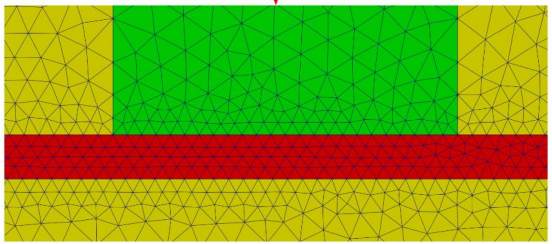
FEM mesh

Gmsh





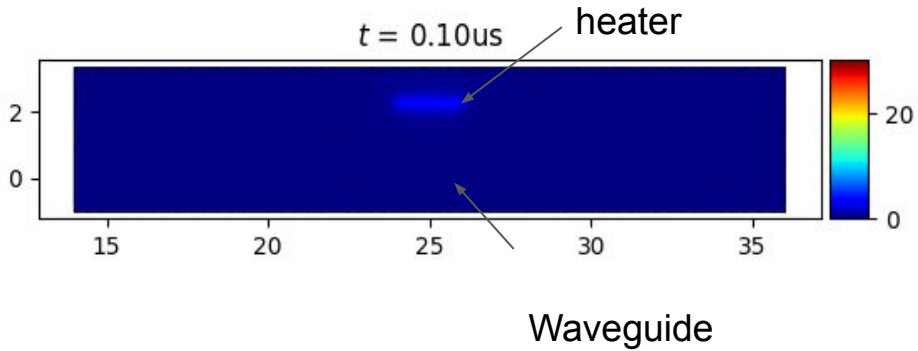


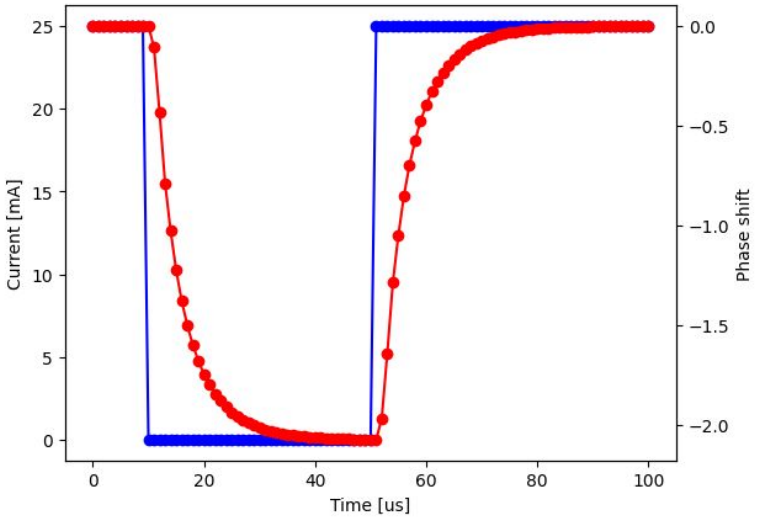


```
waveguide = gf.components.straight_pin(length=10, taper=None)
waveguide.show()

filtered_layerstack = LayerStack(
    layers={
        k: get_layer_stack_generic().layers[k]
        for k in (
            "slab90",
            "core",
            "via_contact",
            # "metal2",
        )  # "slab90", "via_contact")#"via_contact") # "slab90", "core"
    }
)

resolutions = {}
resolutions["core"] = {"resolution": 0.05, "distance": 2}
resolutions["slab90"] = {"resolution": 0.03, "distance": 1}
resolutions["via_contact"] = {"resolution": 0.1, "distance": 1}

geometry = uz_xsection_mesh(
    waveguide,
    [(4, -15), (4, 15)],
    filtered_layerstack,
    resolutions=resolutions,
    background_tag="Oxide",
)
```

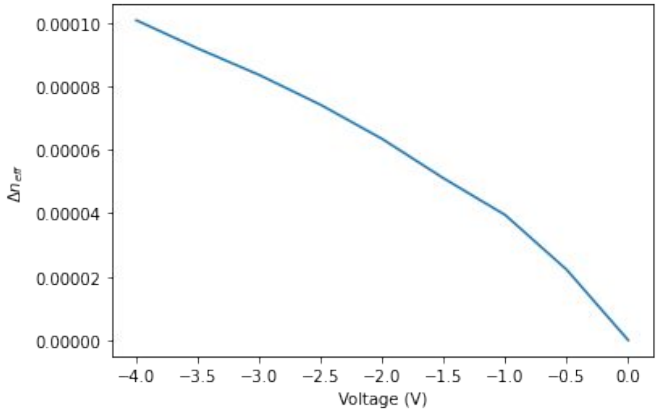
FEM Heat solver

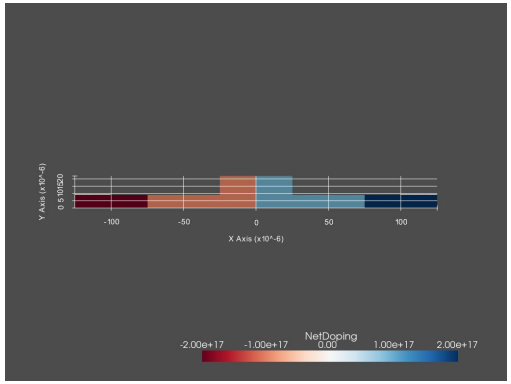
FEMWELL

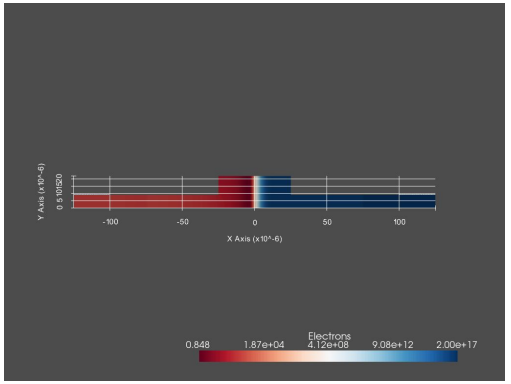


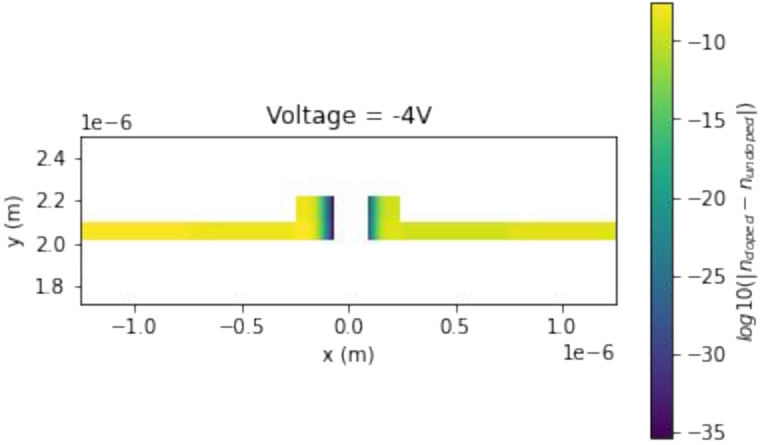


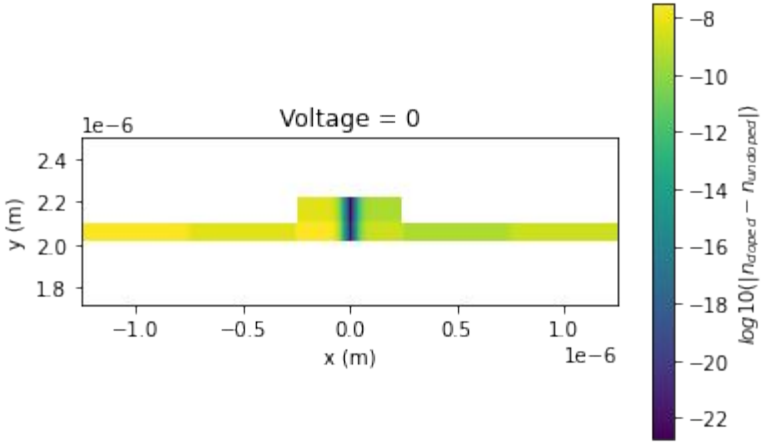
DEVSIM TCAD Device simulator



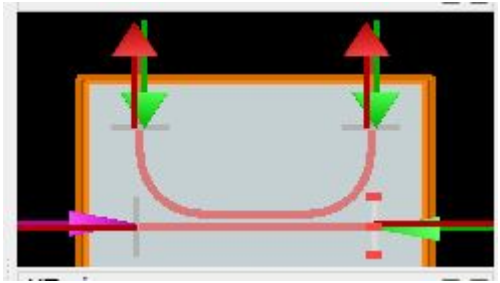


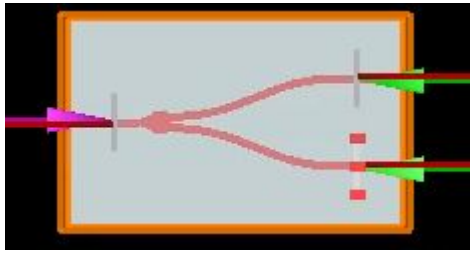


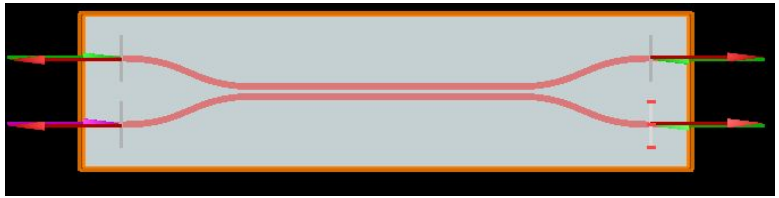


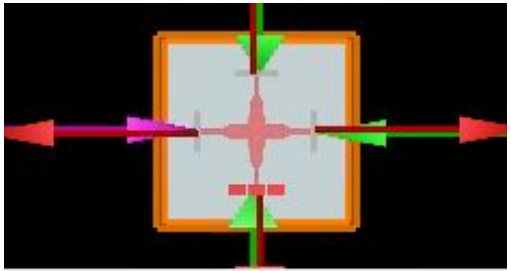


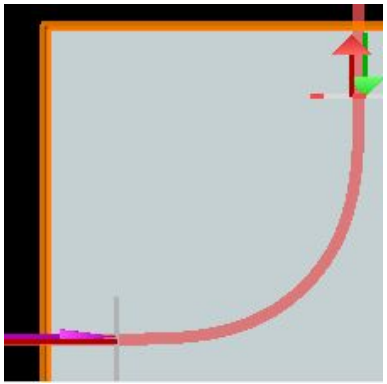
Automated FDTD Simulation from layout

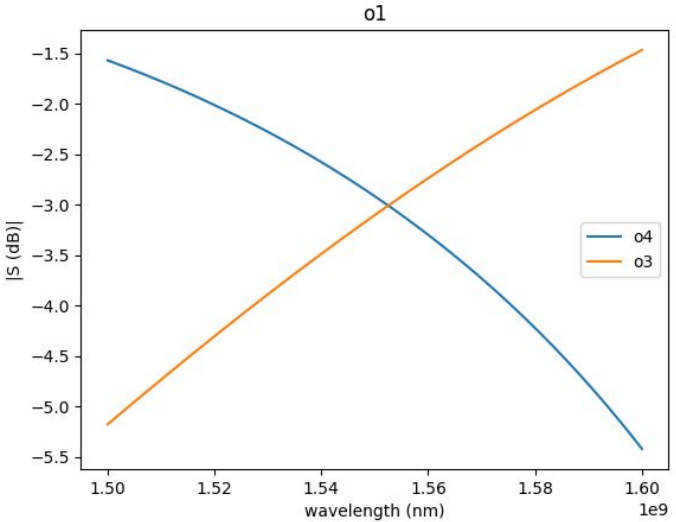




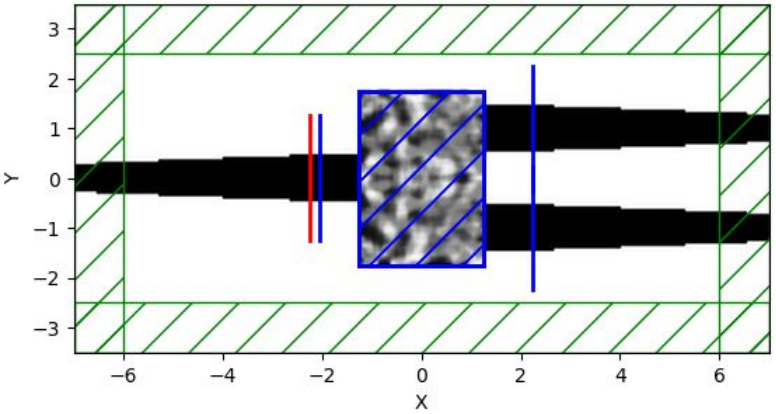


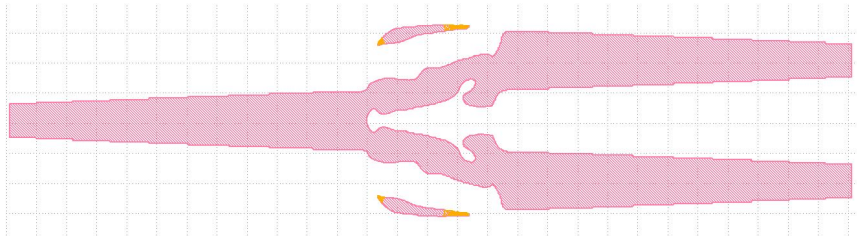


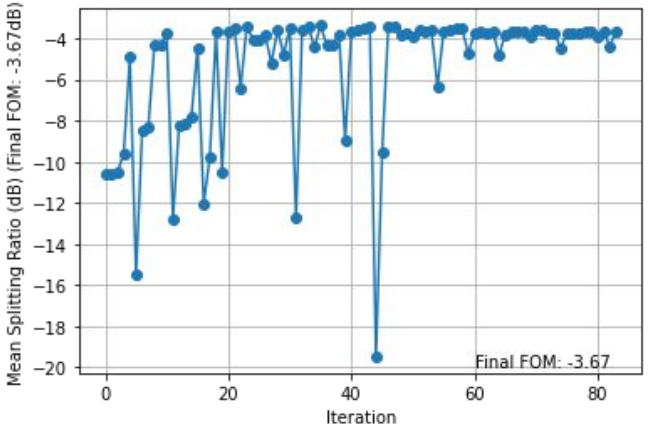


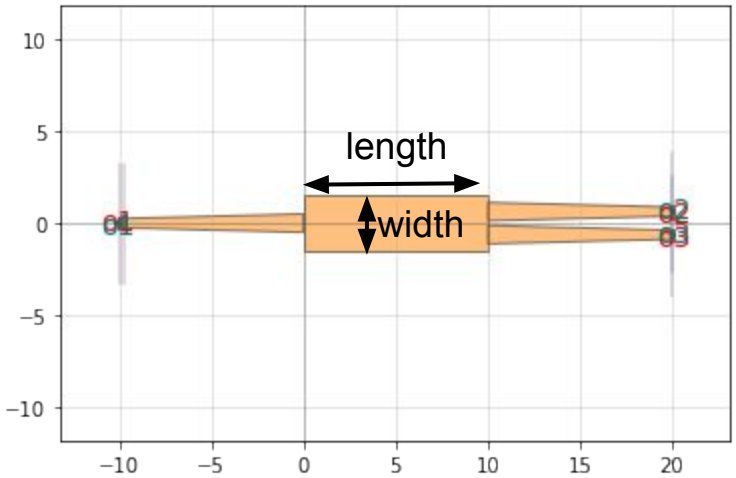


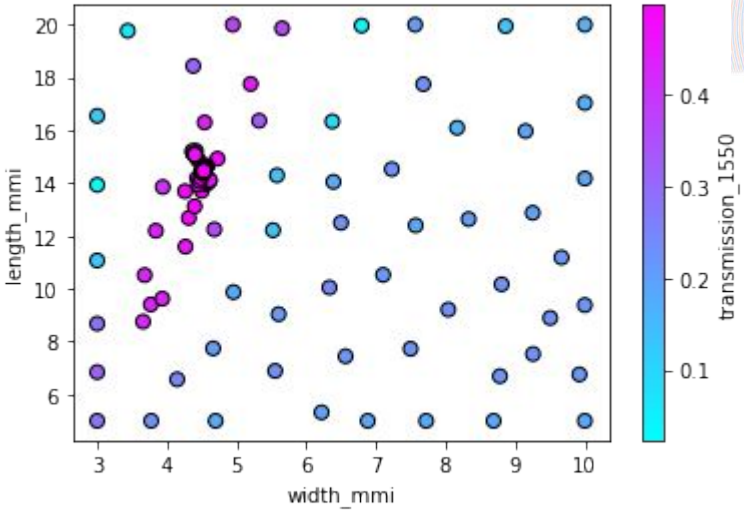
Adjoint optimization (inverse design)

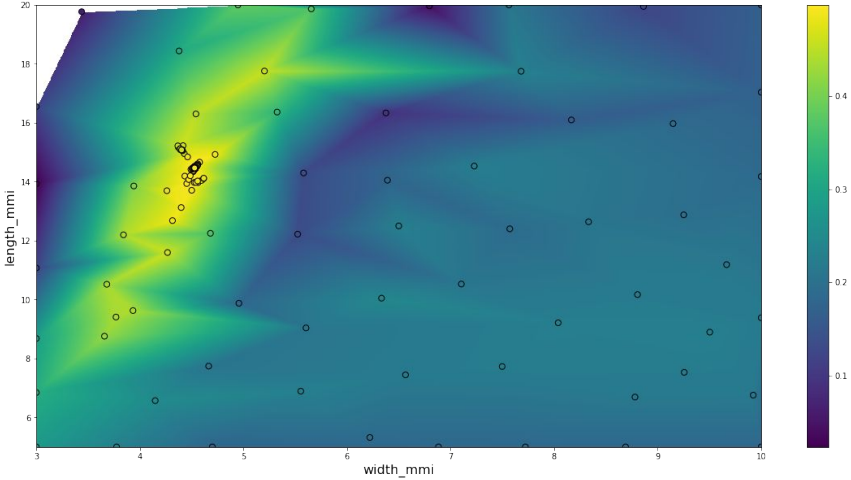






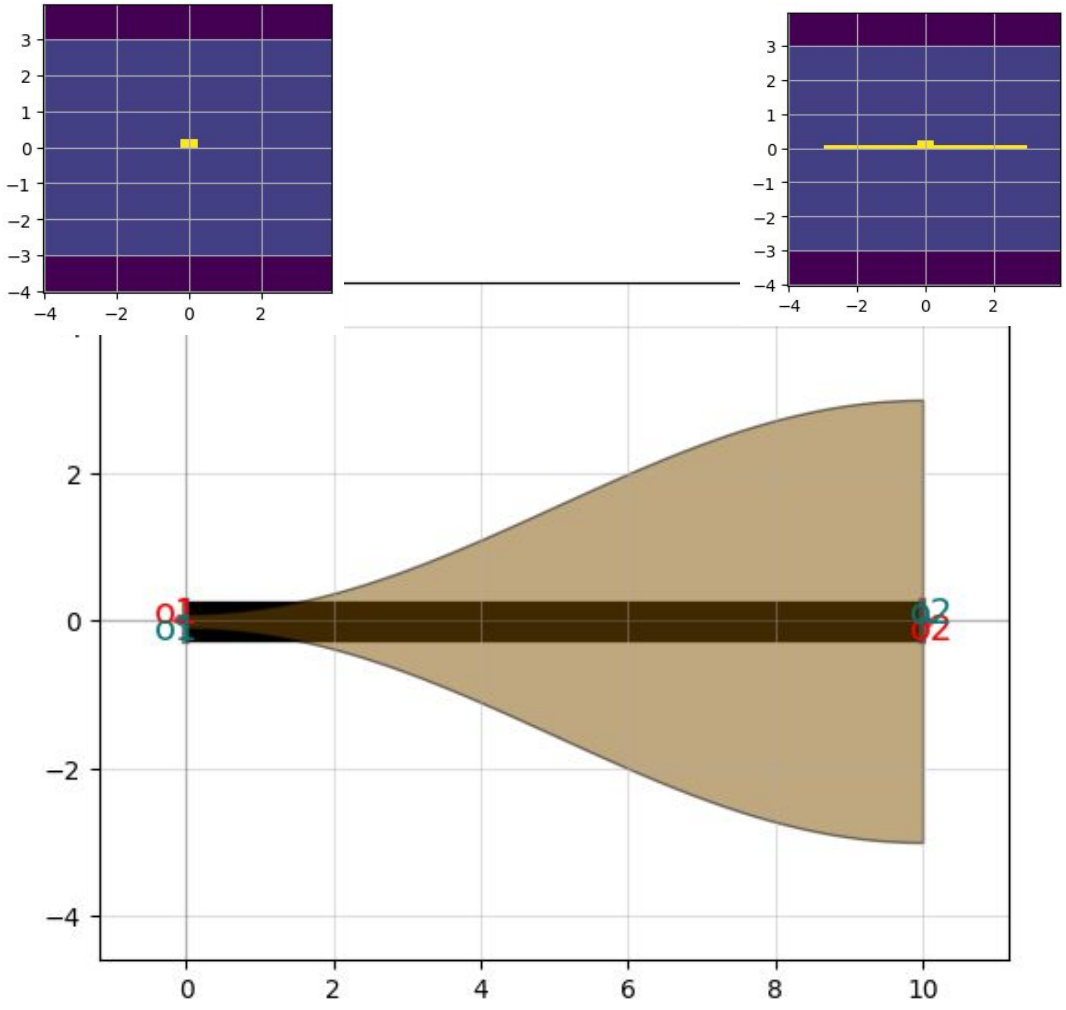




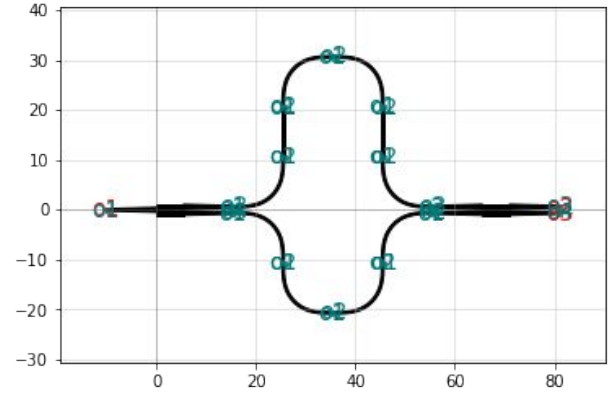


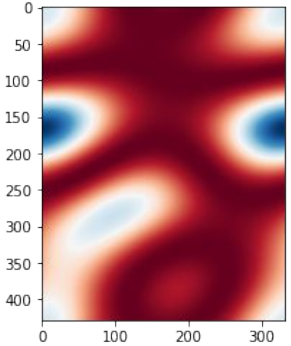
TRIAL_ID	width	length	transmission
128	3.000000	13.936181	0.023320
92	6.796302	19.966311	0.042681
150	4.535979	14.461443	0.497184
148	4.536362	14.458428	0.497189
86	4.531666	14.452401	0.497239
144	4.535423	14.446593	0.497312
104	4.555169	14.544551	0.497325
163	4.535932	14.450433	0.497512
167	4.535889	14.454474	0.497642

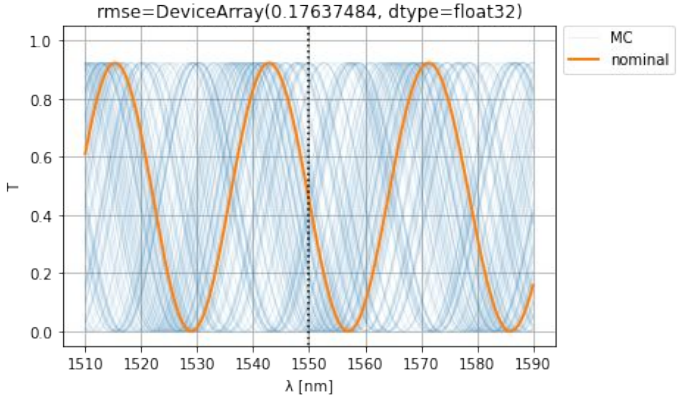
EME

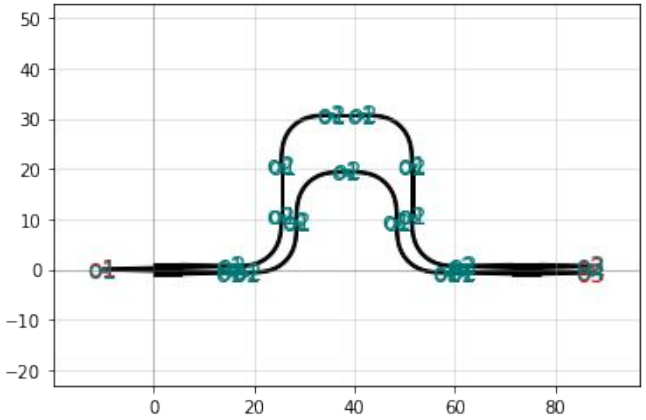


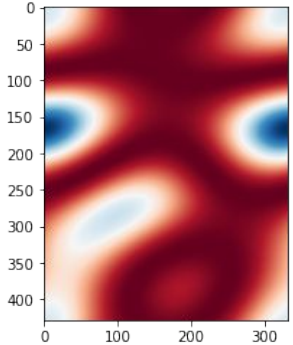
Layout aware Monte Carlo with SAX

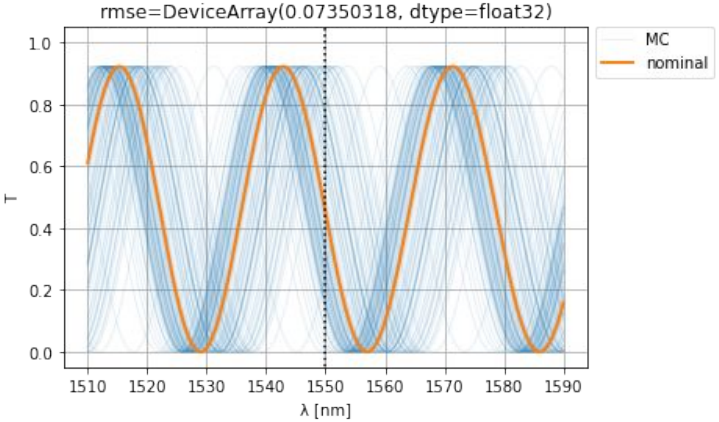


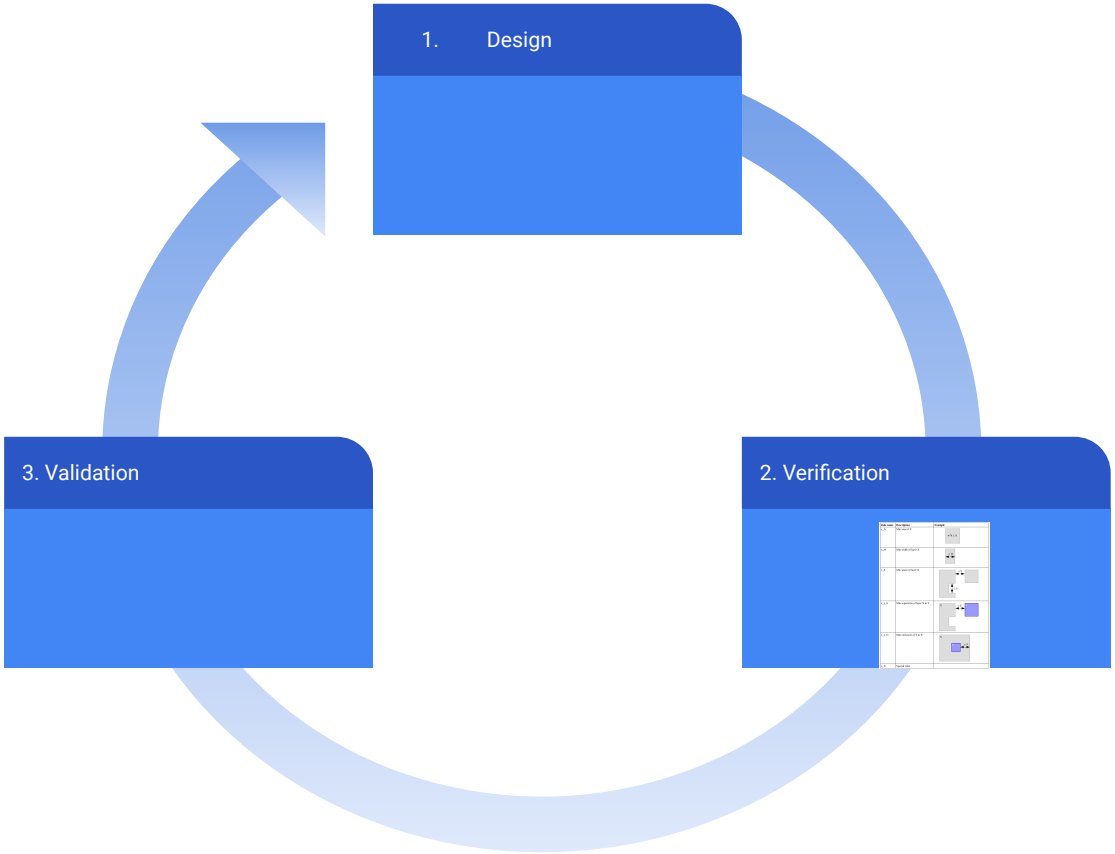










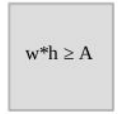
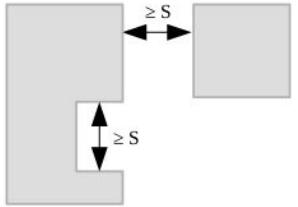
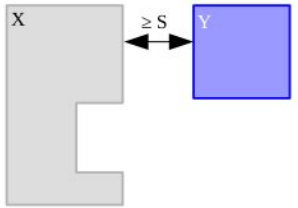
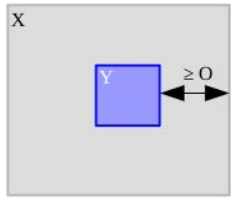

1. Design
2. Verification
3. Validation

DRC

GDSFACTORY → 

Generates Klayout DRC decks and creates shortcut

```
rules = [
    rule_width(layer="WG", value=0.2),
    rule_space(layer="WG", value=0.2),
    rule_width(layer="M1", value=1),
    rule_width(layer="M2", value=2),
    rule_space(layer="M2", value=2),
    rule_separation(layer1="HEATER", layer2="M1", value=1.0),
    rule_enclosing(layer1="M1", layer2="VIAC", value=0.2),
    rule_area(layer="WG", min_area_um2=0.05),
    rule_density(
        layer="WG", layer_floorplan="FLOORPLAN", min_density=0.5, max_density=0.6
    ),
]

drc_rule_deck = write_drc_deck_macro(
    rules=rules,
    layers=gf.LAYER,
    shortcut="Ctrl+Shift+D",
)
```

Rule name	Description	Example
x_A	Min area of X	
x_W	Min width of layer X	
x_S	Min space of layer X	
x_y_S	Min separation of layer X to Y	
x_y_O	Min enclosure of Y in X	
x_X	Special rules	

Database SiEPIC-Tools Verification: generic technology[1] File ...

... on layout mzi.gds

Directory

Cell / Category	Count (Not Visited)
By Cell	7 (7)
By Category	7 (7)
Waveguide	
Component	1 (1)
Connectivity	6 (6)
All	7 (7)

Markers

F	I

Info

[mzi]

Category Cell

Configure Close

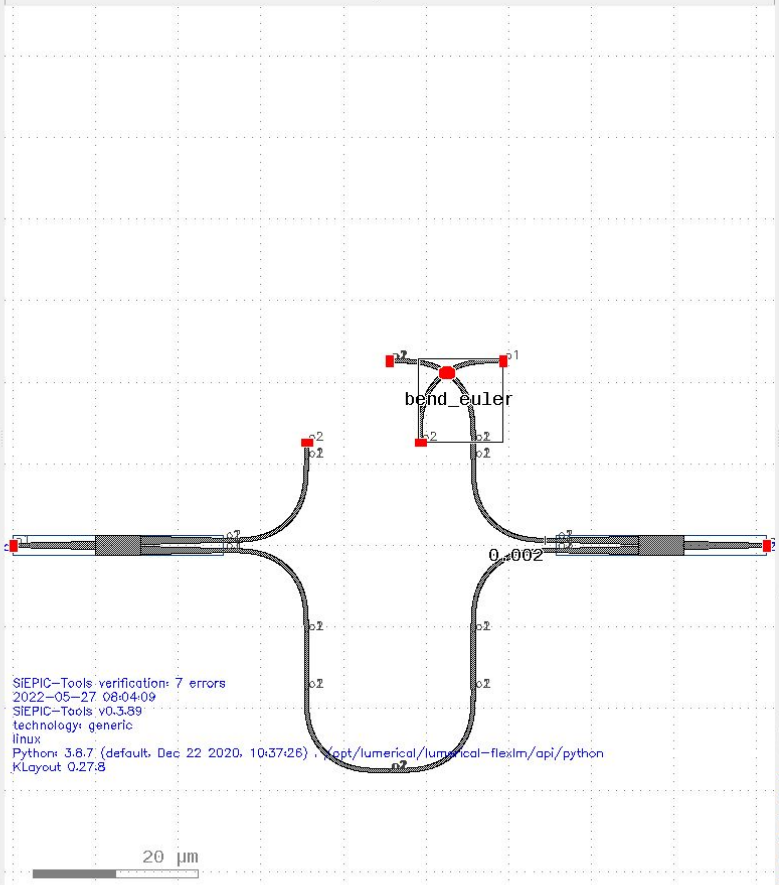
File Edit View Bookmarks Display Tools Macros SiEPIC 0.3.89 Py3 Help generic DRC

Back Forward Select Move Ruler Add Polygon Box Text Path Instance Partial generic Simulation Functional Verification Test Coordinates

Cells

mzi

[+] mzi.gds [mzi]



Levels 0 .. 2

T generic G

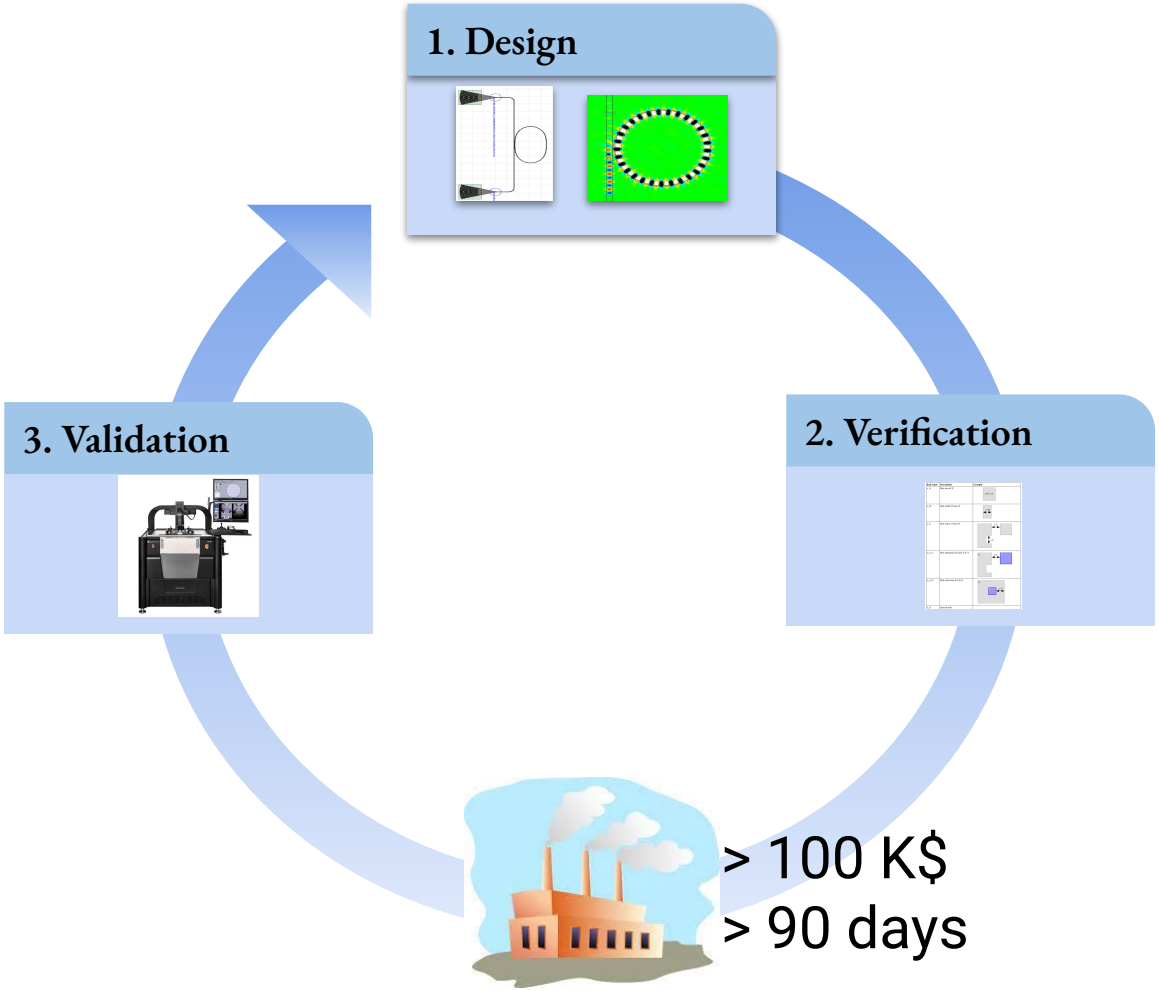
Layers

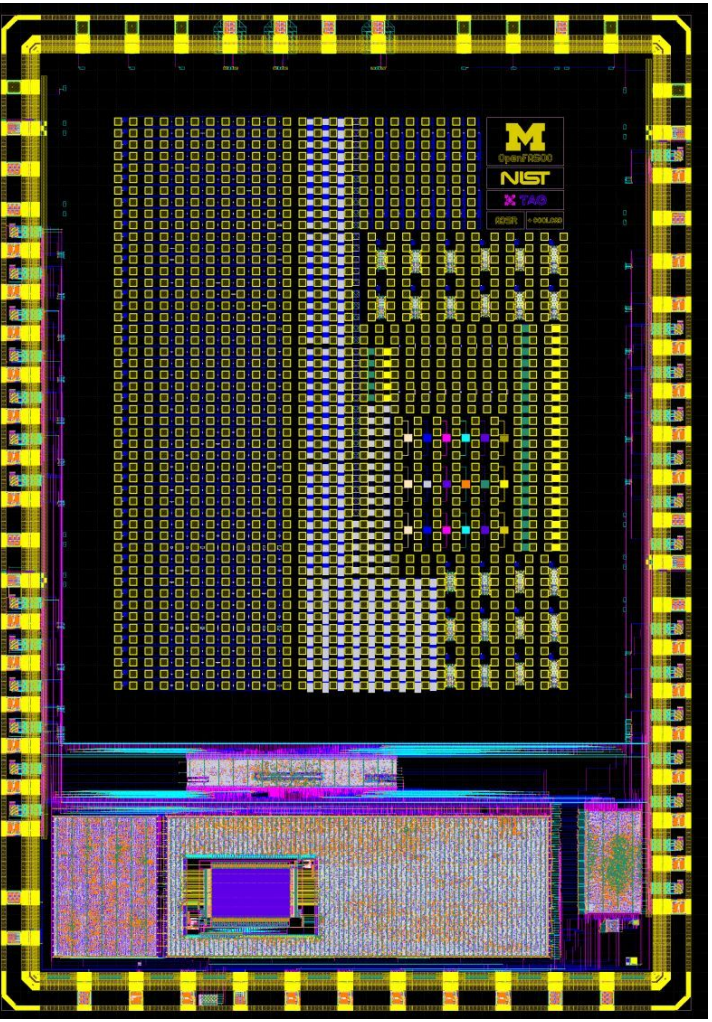
Waveguide
WGCLAD 111/0
SLAB150 2/0
SLAB90 3/0
SHALLOWET...
DEEPETCH 3/6
SLAB150CLA...
SLAB90CLAD...
Doping
WGN Nitride ...
WGNCLAD 3...
GE 5/0
SILICIDE 39/0
MH 47/0
M1 41/0
M2 45/0
M3 49/0
VIAC 40/0
VIA1 44/0
VIA2 43/0
CAPACITOR 4...
METALOPEN ...
DEEPTRENC...
OXIDE_ETCH ...
SITILES 190/0
M1TILES 191/0
TEXT 66/0
LABEL OPTIC...
LABEL_SETTI...
TE 203/0
TM 204/0
LABEL_INSTA...
DICING 65/0
DRC_EXCLUD...
FLOORPLAN ...
simulation
Lumerical
DevRec
PinRec
FbrTgt
Text
Errors
PinRecM

Layer Toolbox

Color
Frame color
Stipple
Animation
Style
Visibility

xy -10.46404 29.68868


1. Design
2. Verification
3. Validation
> 100 K$
> 90 days



Tuohang Zeng • 1st
MS ECE student at the University of Michigan
6mo • Edited •

Tape-out submitted! I am excited to complete my first tape-out with **Mehdi Saligane** as my advisor at the **University of Michigan**. This work was done in collaboration with **Brian Hoskins** at the **National Institute of Standards and Technology (NIST)**, **David Fleischer** at **ADSR, Ltd.**, and **Akin Akturk** at **CoolCAD Electronics LLC**. Special thanks to **Tim Ansell** and **Google** for sponsoring the **SkyWater Technology Foundry**, **Google**, **Efabless Corporation** MPW program!

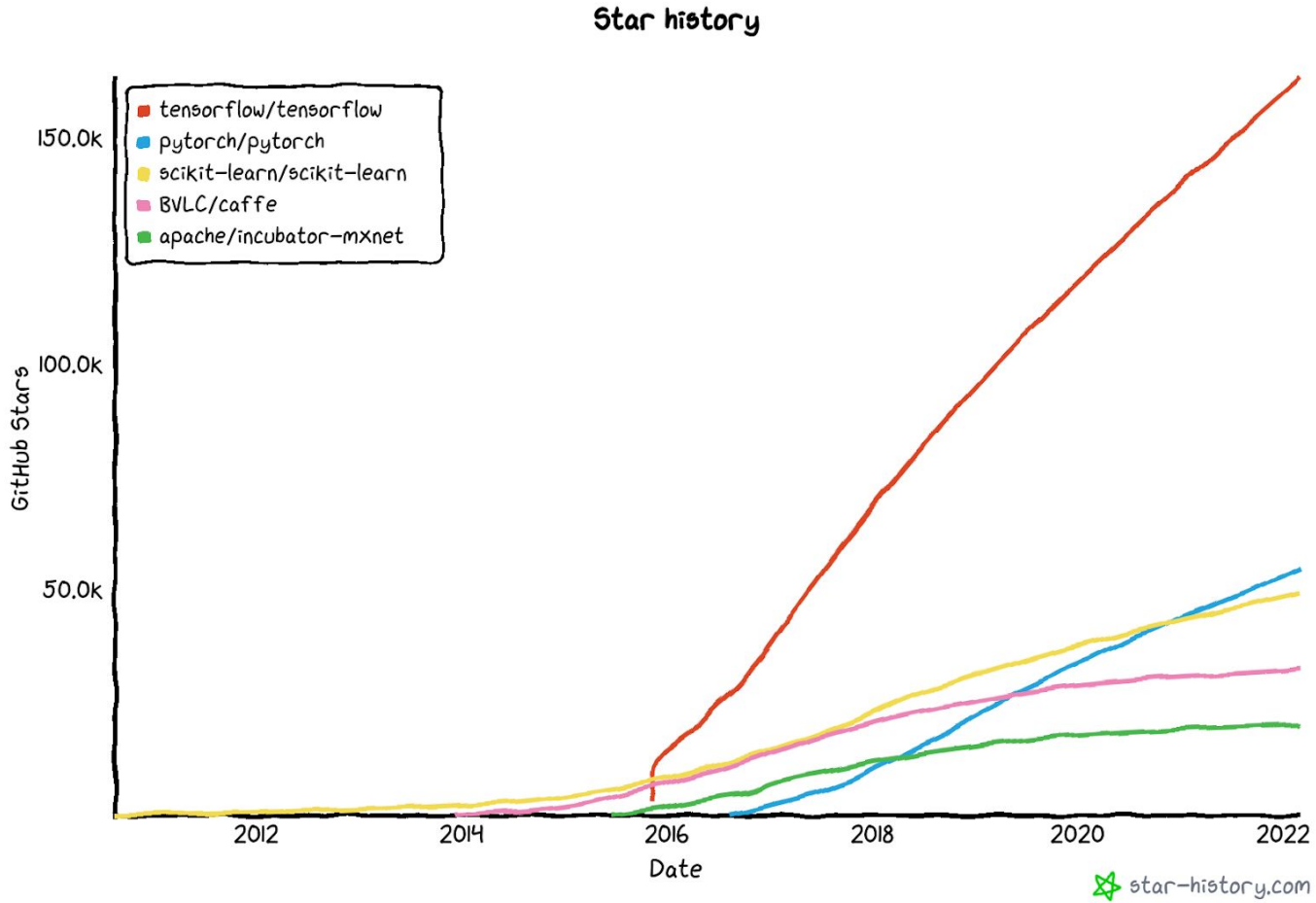
Our design consists of test structures on the open-source SKY130 PDK. Few stats to highlight:

- ~1400 bare pads
- 400+ transistor test structures
- 30 capacitor test structures
- 24 Ring Oscillators, built on 12 standard cell libraries using OpenFASOC
- 18 line resistance and via chain test structures
- ...

We completed our designs using open-source tools like Magic and KLayout, and heavily used automated flows like OpenFASOC, sitting on top of OpenROAD, and gdsfactory. The goal is to create open-source models at cryogenic temperature and enhance the existing SKY130 models, especially for high-end analog design.

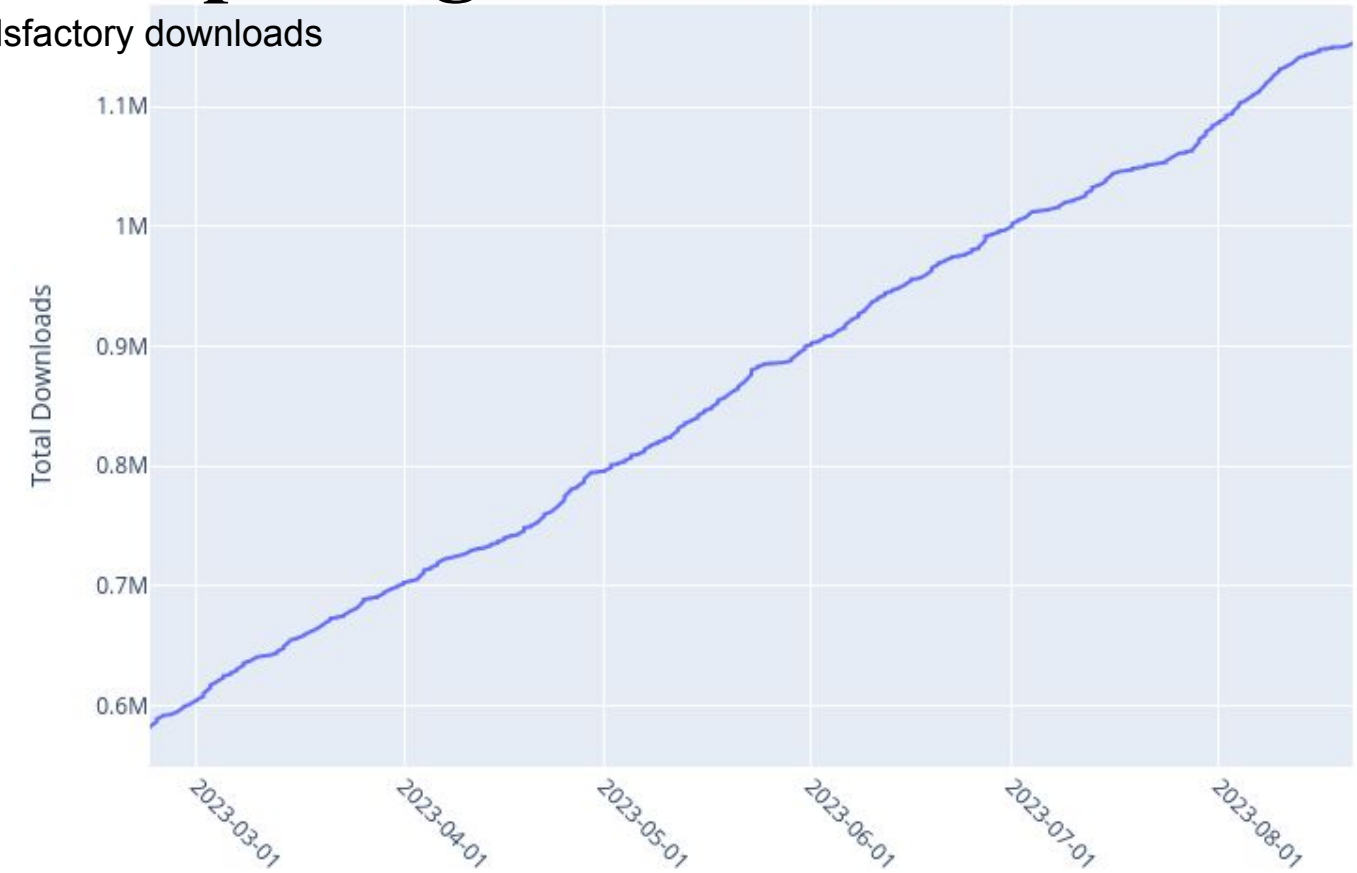
Open-source hardware/EDA has enabled a new level of collaboration in IC design, and we are excited about what open-source PDKs and tools will empower us to do in the future!

Bring the **success** of machine learning



Into **Chip design**

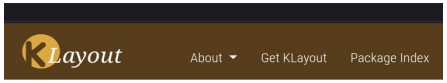
Gdsfactory downloads



Getting started

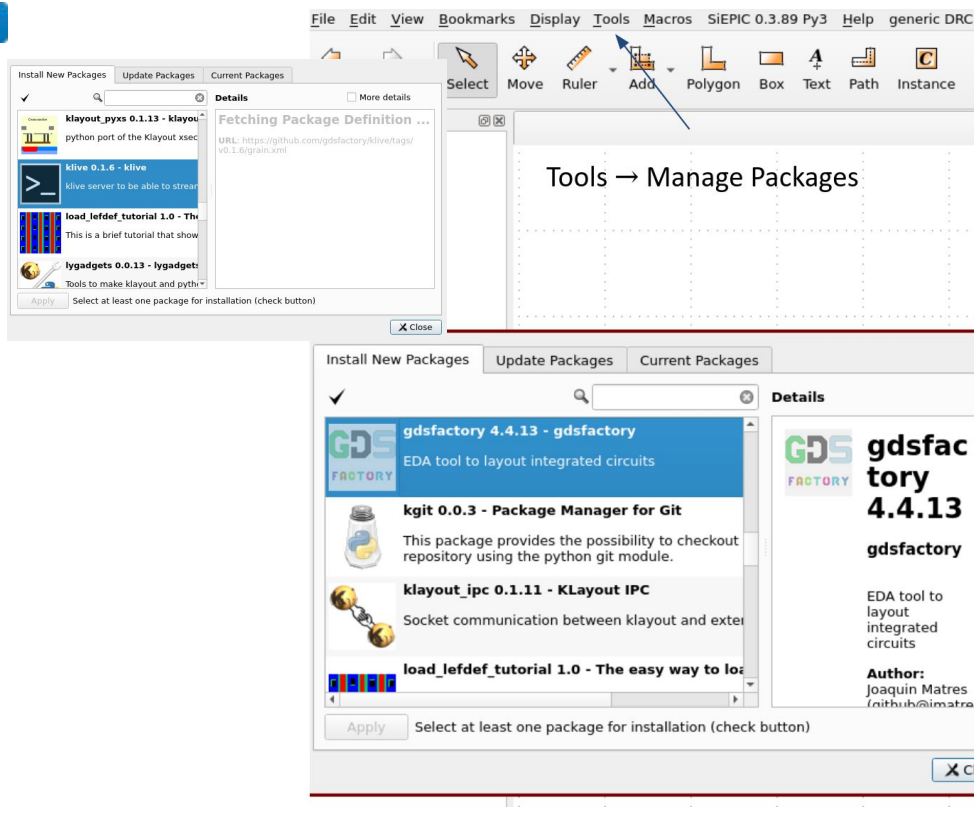
- Learn on Google Colab 

- Install it on your computer
  - Install anaconda
  - Install klayout
  - Install gdsfactory macro and Klive
- See also
  - YouTube videos
  - Gdsfactory docs
  - Gdsfactory plugins docs
  - Photonics edx course



Download or Build Yourself

Download




```python
@gf.cell
def mzi(delta_length: float = 10.0) -> gf.Component:
    """Create a Mach-Zehnder Interferometer component with an adjustable length difference
between the two arms.

    Args:
        delta_length: Length difference between the two arms of the MZI.

    Returns:
        Mach-Zehnder Interferometer component.
    """
    c = gf.Component()

    splitter = c << gf.components.mmi1x2()

    # Bottom arm
    bottom_arm = c << mzi_arm()

    bottom_arm.mirror(p1=(0, 0), p2=(1, 0))

    bottom_arm.connect(port="o1", destination=splitter.ports["o3"])

    # Combiner
    combiner = c << gf.components.mmi1x2()
```