GDSFactory: An Open-Source Python Library for Chip Design and Simulation

Joaquin Matres,[1,*] Simon Bilodeau,[3] Niko Savola,[2,4] Marc de Cea,[2] Wai Kwan Yeung,[5] Erman Timurdogan,[7] Helge Gehring,[6] and Troy Tamas[1]

[1]*GDSFactory, 650 Castro St Ste 120 PMB 98035, Mountain View, CA 94041, USA*
[2]*Taara Co., Mountain View, CA, USA*
[3]*Department of Electrical and Computer Engineering, Princeton University, Princeton, NJ 08544, USA*
[4]*Department of Applied Physics, Aalto University, PO Box 13500, FIN-00076 Aalto, Finland*
[5]*Department of Physics, University of Oxford, Oxford, UK*
[6]*Google LLC, 1600 Amphitheatre Parkway, Mountain View, CA 94043, USA*
[7]*Lumentum, San Jose, CA, USA*

[*]jmatres@gdsfactory.com

Abstract: We present GDSFactory, an open-source Python library for integrated circuit design automation supporting photonics, analog, quantum, and MEMS applications. The platform provides unified workflows spanning layout design, device simulation through FDTD (MEEP, Tidy3D, Lumerical), FEM (Femwell), and TCAD (DEVSIM) solvers, and circuit simulation via SAX S-parameter analysis.

1 Introduction

Hardware iterations typically require months and millions of dollars, while software iterations cost hundreds of dollars and take hours. GDSFactory bridges this gap by providing a comprehensive Python API for chip development, including layout design, device simulation, circuit simulation, and verification [1].

Unlike constrained logic-driven electronic design flows [2], integrated photonics requires freeform parametric geometries and tight integration between layout and electromagnetic simulation. GDSFactory addresses this with scriptable parametric cells (PCells), hierarchical assembly, routing, and seamless interfaces to industry-standard simulators—all in Python's extensive scientific ecosystem (Fig. 1).

Figure 1. GDSFactory workflow: Design phase (simulation, schematic capture, layout), Verification (DRC, LVS), and Validation to ensure the fabricated product meets specifications.

2 Layout Design

GDSFactory implements cells as Python functions returning a `Component` class with polygons, port metadata, and convenience methods. Using KLayout's C++ geometry engine [3], users define PCells with a functional programming approach:

```
import GDSFactory as gf

@gf.cell
def mzi_with_bend(radius=10):
    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'])
    return c
```

The `@gf.cell` decorator handles caching, eliminating redundant regeneration. Port metadata enables automatic routing via `get_route` and `get_bundle` functions that connect components following `CrossSection` specifications (Fig. 2).

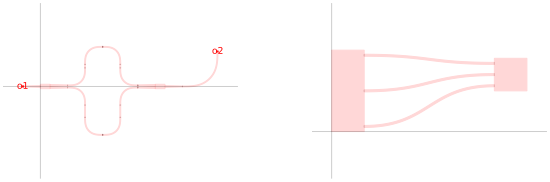

Figure 2. (a) MZI with Euler bend showing port connections. (b) Routed $n \times n$ components using S-bends.

3 Device Simulation

GDSFactory's gplugins repository provides unified interfaces to device simulators by reusing layout abstractions. Components can be meshed via GMSH for cross-sectional or 3D analysis (Fig. 3).

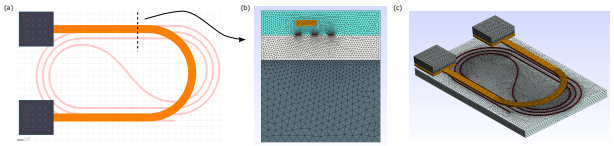

Figure 3. GDSFactory meshing: (a) heater layout, (b) cross-sectional mesh, (c) 3D mesh.

Full-wave electromagnetic simulation is supported through three FDTD backends: MEEP [4] (open-source), Tidy3D [5] (cloud-based GPU acceleration), and Lumerical FDTD. FEM and multiphysics solvers include Femwell [6] for waveguide mode analysis and thermal simulation, Palace [8] for RF and microwave simulations, MPB for photonic crystals, and MEOW for eigenmode expansion. TCAD simulation uses DEVSIM [7] for semiconductor device physics and Sentaurus for advanced process simulation.

4 Circuit Simulation

Circuit-level simulation enables system-scale photonic design through netlist extraction and S-parameter composition. SAX [9] provides differentiable S-parameter circuit simulation using JAX, supporting automatic differentiation for gradient-based optimization, Monte Carlo analysis for yield estimation, and wavelength-dependent S-parameter interpolation from FDTD results. VLSIR provides SPICE netlist export for mixed photonic-electronic simulation.

5 Process Design Kits

Open-source PDKs include GlobalFoundries 180nm, SkyWater 130nm [10], VTT 3 µm SOI, and SiEPIC. Commercial PDKs under NDA include AIM, AMF, TowerSemi, IMEC, and HHI. The generic PDK follows standard layer conventions [11] for cross-foundry compatibility.

6 Conclusion

GDSFactory provides a unified Python-driven workflow spanning layout design, device simulation, and circuit simulation. The tight integration between FDTD solvers (MEEP, Tidy3D, Lumerical), FEM tools (Femwell, DEVSIM), and circuit simulators (SAX) enables rapid design iteration from component to system level. The library is freely available at https://github.com/GDSFactory/GDSFactory.

References

[1] J. Matres *et al.*, "GDSFactory," GitHub (2024), https://github.com/GDSFactory/GDSFactory.

[2] W. Bogaerts *et al.*, "Silicon photonics circuit design: methods, tools and challenges," Laser Photon. Rev. **12**, 1700237 (2018).

[3] M. Köfferlein, "KLayout," https://www.klayout.de/.

[4] A. F. Oskooi *et al.*, "MEEP: A flexible free-software package for electromagnetic simulations by the FDTD method," Comput. Phys. Commun. **181**, 687–702 (2010).

[5] Flexcompute Inc., "Tidy3D," https://www.flexcompute.com/tidy3d/.

[6] H. Gehring *et al.*, "Femwell," GitHub (2023), https://github.com/HelgeGehring/femwell.

[7] J. E. Sanchez, "DEVSIM," https://devsim.org/.

[8] AWS, "Palace: 3D Finite Element Solver for Computational Electromagnetics," GitHub (2024), https://github.com/awslabs/palace.

[9] F. Laporte, "SAX," GitHub (2023), https://github.com/flaport/sax.

[10] SkyWater Technology Foundry and Google, "SkyWater Open Source PDK," GitHub (2023), https://github.com/google/skywater-pdk.

[11] L. Chrostowski and M. Hochberg, *Silicon Photonics Design* (Cambridge, 2015).