Andreas Krinke
Institute of Electromechanical and Electronic Design, Dresden University of Technology

Layout Verification Using Open-Source Software

International Symposium on Physical Design
Taipei, March 14, 2024

andreas.krinke@tu-dresden.de
www.ifte.de

Outline

1. Motivation
    - Open-Source Software for Layout Verification
    - OpenPDKs
2. Generating DRC/LVS Runsets for KLayout
    - Data Structure
    - KLayout Generator
    - Current Status
3. Conclusion & Outlook

Motivation: Problems with Commercial EDA Tools

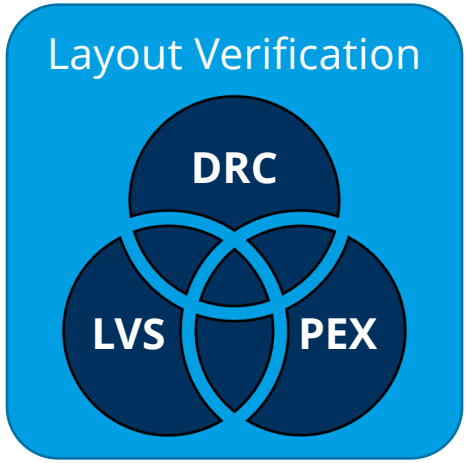
- Licensing Costs
  - High licensing costs, higher for smaller technology nodes
  - Discounts, maybe?

- Confidentiality
  - Example: Trustworthy electronics (e.g. for cryptography)
  - For software: Reproducible builds – from source to binary
  - For integrated circuits: mask layout is confidential

Motivation: Layout Verification

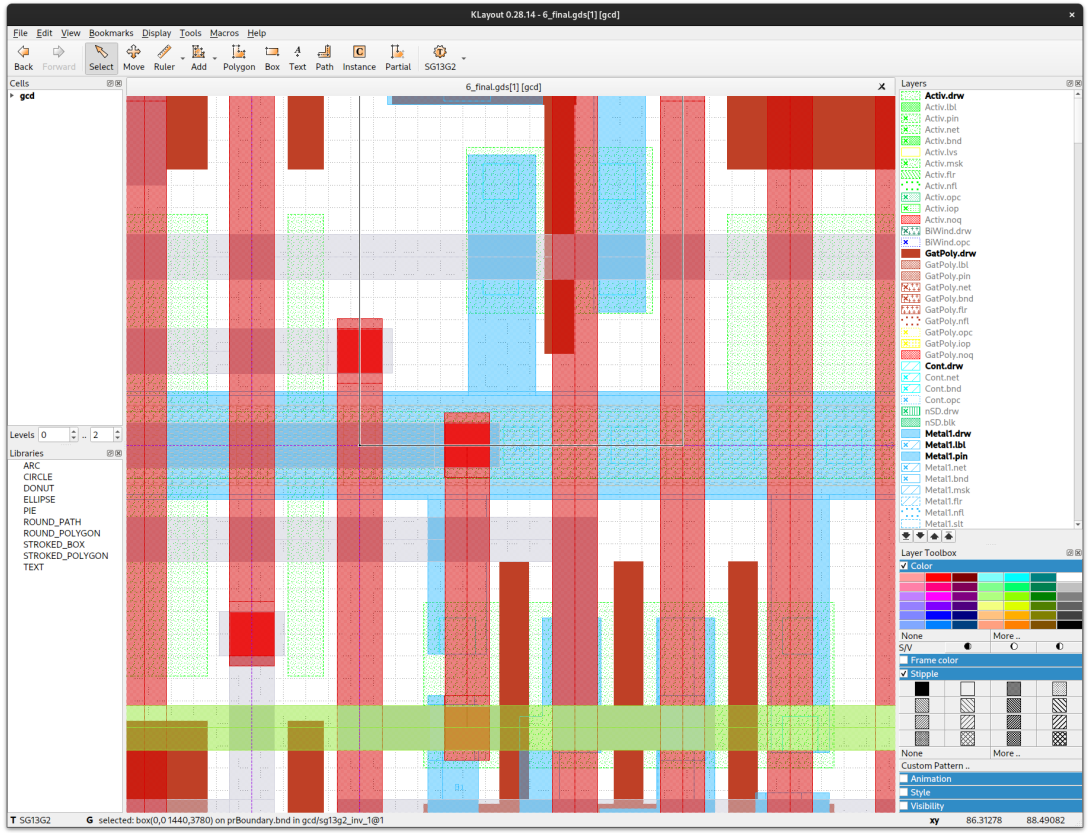
- Before production: Verification of mask layouts
  - Design Rule Check **DRC**
  - Layout vs. Schematic **LVS**
  - Parasitic Extraction **PEX**
- Technology information:
  - Part of a foundry's **PDK**
  - Available in proprietary formats (e.g. SVRF)
  - Again: High licensing costs for required software tools

- Our goal:
  Lower barriers to entry for smaller companies – How?

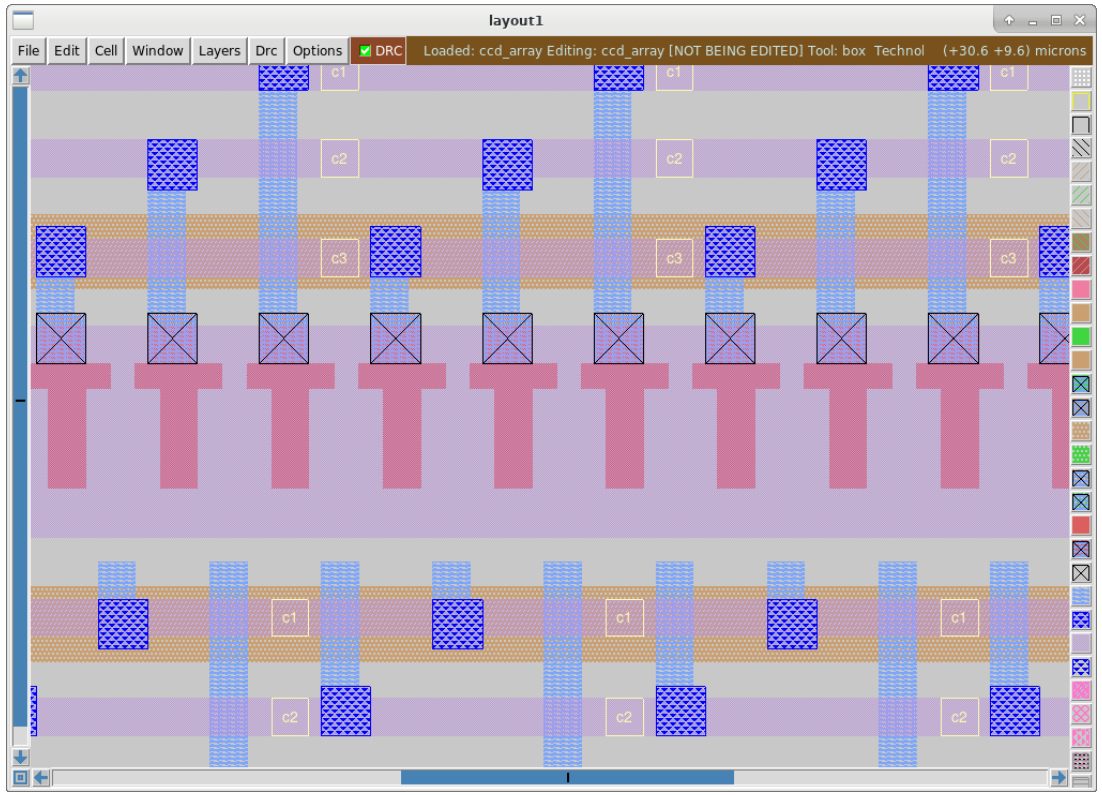


Open-Source Software for Layout Verification

KLayout: DRC, LVS



Magic VLSI: DRC, LVS, PEX



PEX: **FastFieldSolvers** (FastCap, FastHenry, ...), **OpenRCX**

AG EDA ifte

OpenPDKs

- Problem: Open-source tools require **OpenPDKs**

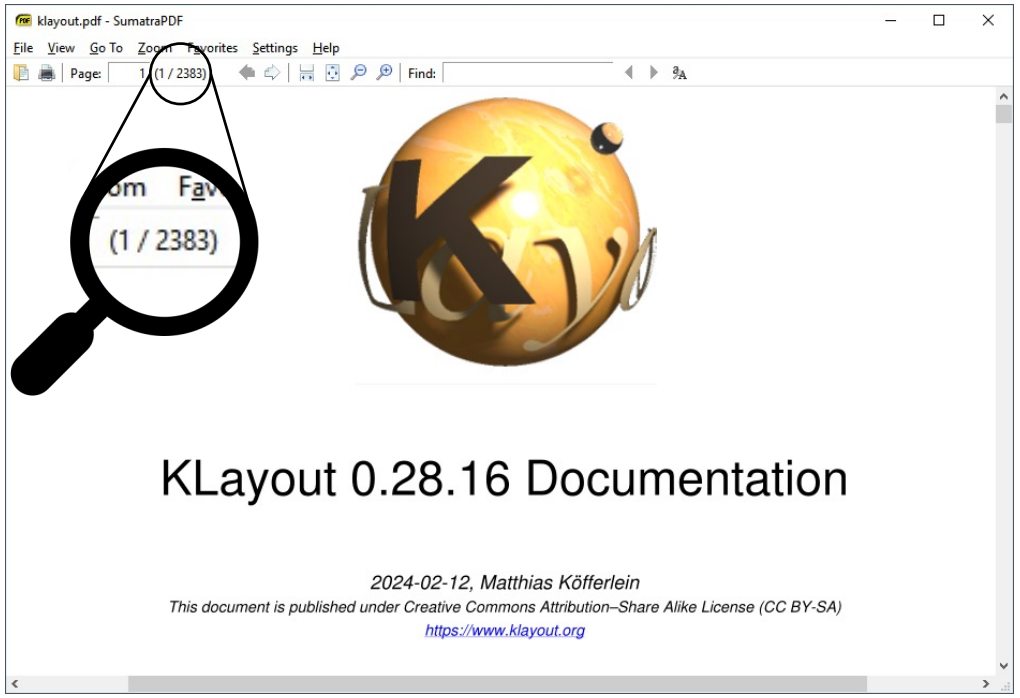
- OpenPDK: PDK that supports open-source tools; *independent of its license*
- Examples:
  - SkyWater 130 nm
  - GlobalFoundries 180 nm

- Our goal: **Generating DRC and LVS runsets for KLayout**

Why?

Motivation: Why KLayout?

- Author: Matthias Köfferlein (+ contributors)
- First official release in 2006
- **> 500 k LOC** (C++); estimated cost: **$24 M**
- KLayout DRC/LVS scripts are written in **Ruby**
  - Support for many typical DRC operations:
    antenna checks, density, connectivity, …
  - Extensible
- Support for parallelization
- Comprehensive documentation
- **Strong copyleft license (GPL)**



In theory, no limits for what we can achieve

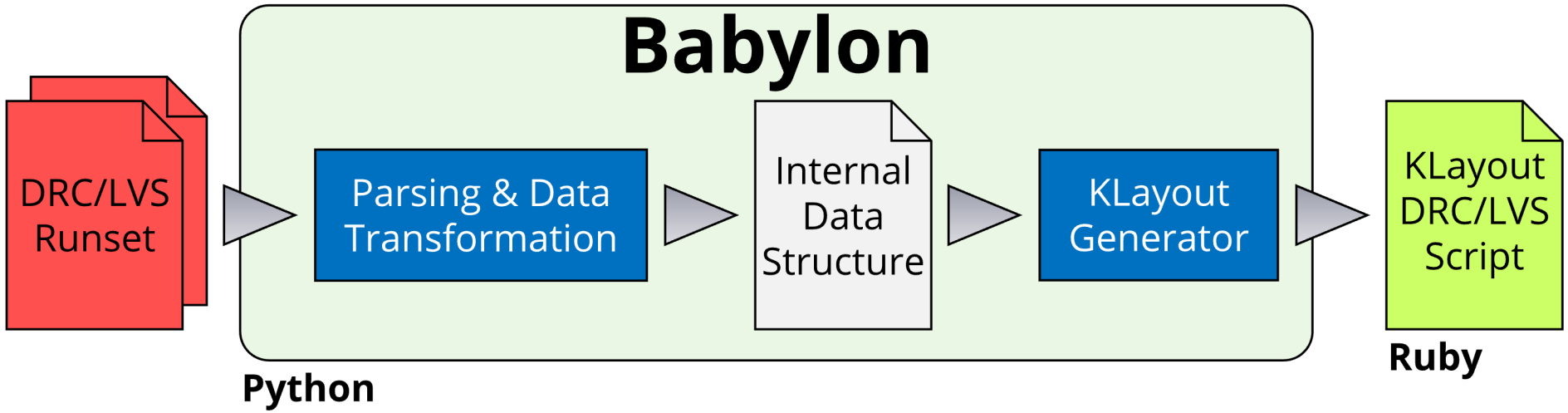
AG EDA ifte

Motivation: Why Generate?

If we would write the KLayout DRC/LVS runset by hand …

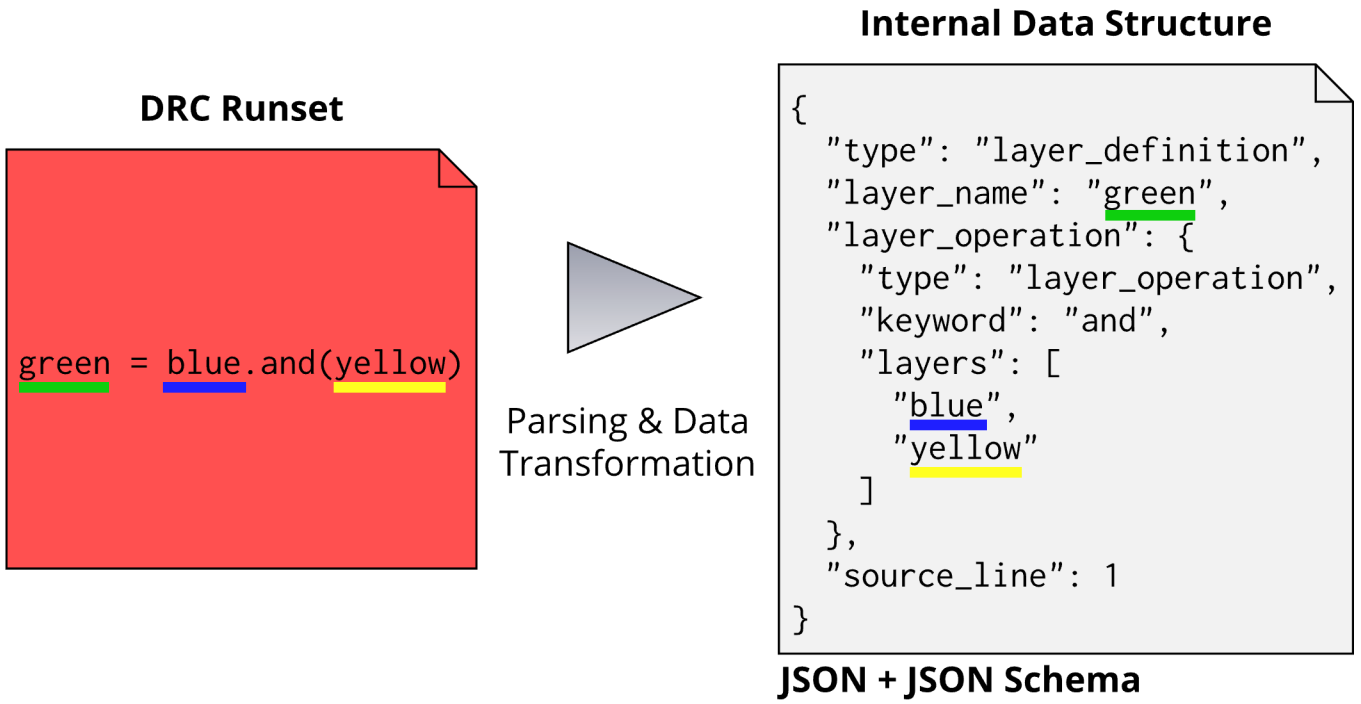
- Large variety of DRC commands → extensive KLayout scripts
- Possibly a lot of code for “simple” checks
- Mitigation:
  - Custom functions and methods
  - Modularization
- Still: Great effort for new technologies

Our Approach

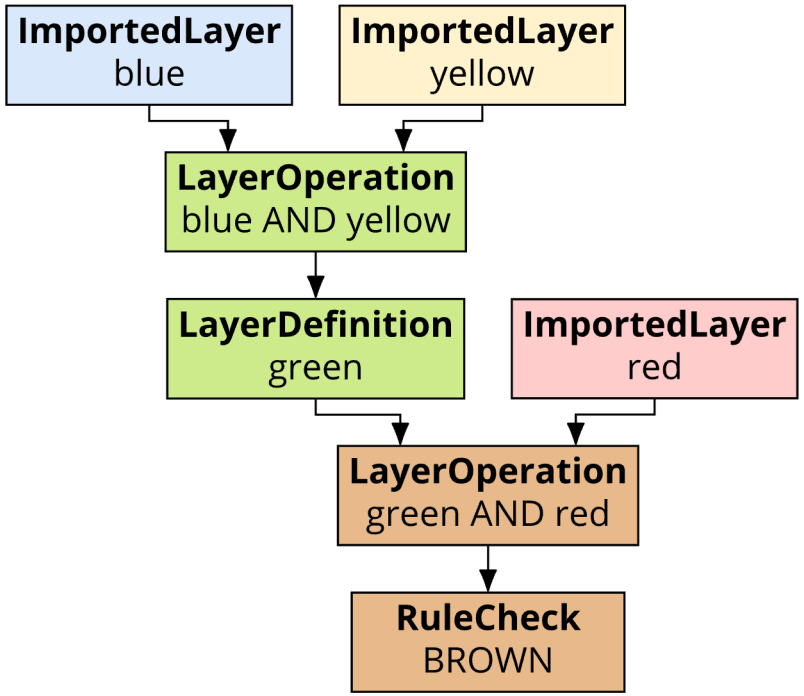


- Reference technology: **X-FAB XH018 180 nm** technology
- Input format: KLayout script (Ruby) with (1) extended API, and (2) irrelevant command order
- Target format: KLayout DRC/LVS script (Ruby)

(Internal) Data Structure



KLayout Generator: Internal Data Model



- Can an object be represented in KLayout?
- Are all required arguments available?
- Is an object part of a design rule?

KLayout Generator: Simple things are simple

Layer assignments

```
name    = input(number)
blue    = input(1)
```

Layer definitions

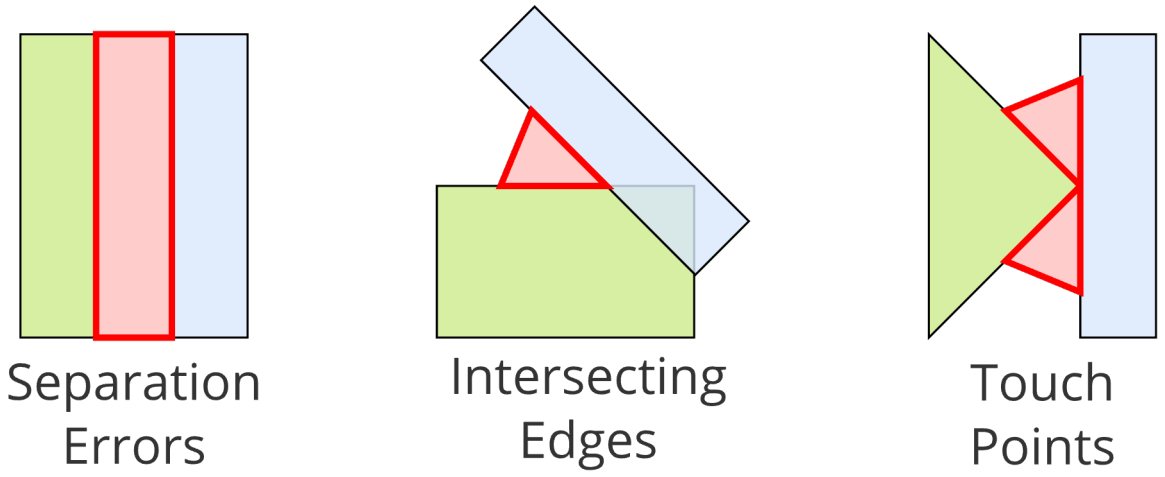
```
name    = layer operation
green   = blue.and(yellow)
```

Design rules

```
(layer operation).output(rule name, comment)
(green.and(red)).output("BROWN", "Is that chocolate?")
```

KLayout Generator: Complicated things are possible

Types of Two-Layer Separation Errors



Ruby / KLayout

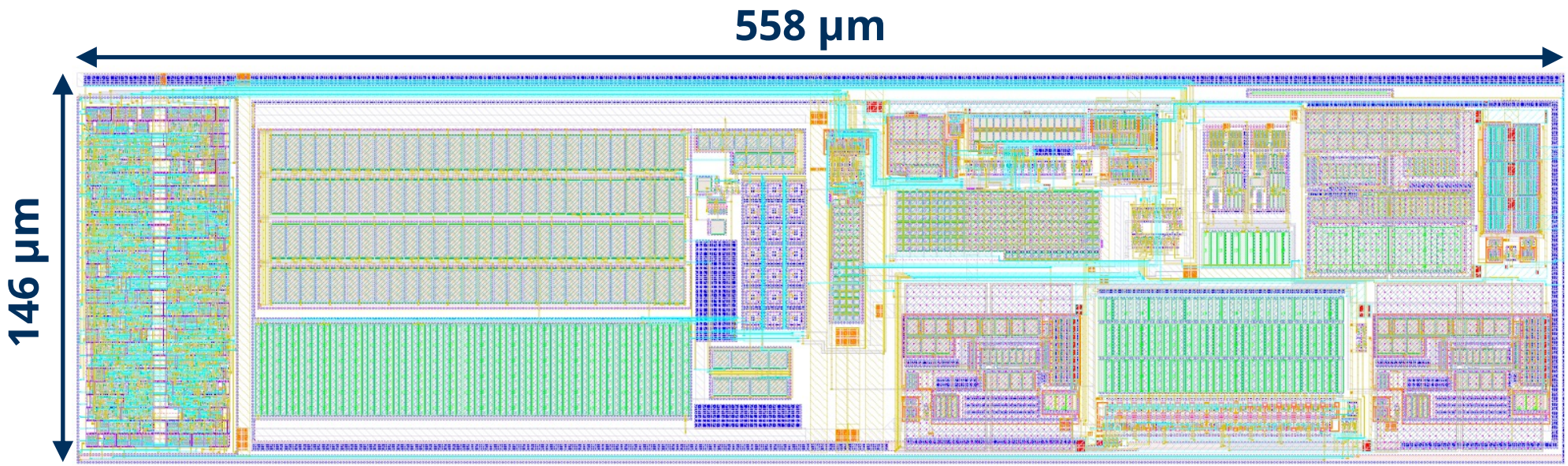
```
class DRC::DRCLayer
    def ext_separation(other, value, ...)
        separation_errors = ...
        intersecting_edges_errors = ...
        touch_point_errors = ...
        return (separation_errors
            + intersecting_edges_errors
            + touch_point_errors)
    end
end

blue.ext_separation(green, 1.0).output(...)
```

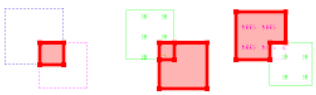

AG EDA ifte

Verification on Test Layouts

- Behavior of layer operations verified on test layouts
- Partial or full support of **33 layer operations** from the design manual

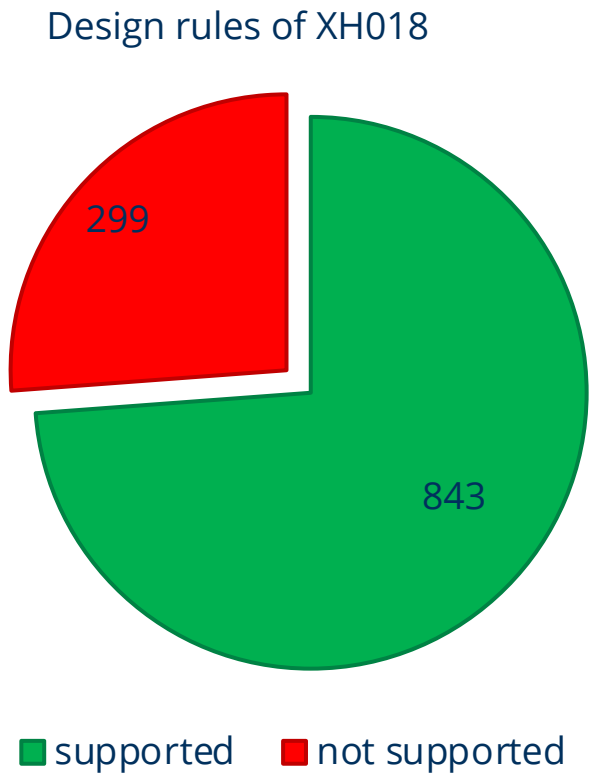


Only intended DRC errors



Current Status

Support for **74%** of the 1142 design rules of XH018



Conclusion

- Comprehensive DRC *and* LVS for commercial technologies using KLayout
- Including, e.g.
  - Marker Browser
  - Creation of result databases (RDBs)
- **Input:** Extended KLayout script (Ruby)
- **Output:** Vanilla KLayout DRC/LVS script (Ruby)

- Our generator is not open source; output can be part of an OpenPDK

Outlook

OpenPDK for IHP's SG13G2 130 nm Technology

- BiCMOS technology, high bipolar performance with f_T/f_{max} = 350/450 GHz
- Our contribution: KLayout DRC script

EM-DRC

- Goal: Electromigration check based on new stress-based EM models
- Inputs: Currents, interconnect geometries
- Tool: KLayout

Outlook

OpenPDK for IHP's SG13G2 130 nm Technology

- BiCMOS technology, high bipolar performance with f_T/f_{max} = 350/450 GHz
- Our contribution: KLayout DRC script

EM-DRC

- Goal: Electromigration check based on new stress-based EM models
- Inputs: Currents, interconnect geometries
- Tool: KLayout

Thank you!

Current Work: (1) LVS

- **LVS = Layout Versus Schematic**
- **4 major steps:**
  - Device recognition
  - Device parameter calculation
  - Connectivity extraction
  - Netlist comparison

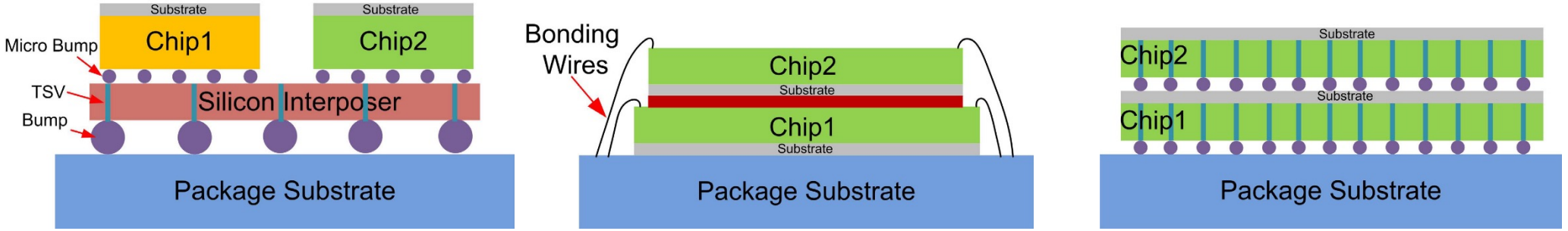
- **Support of new device types requires**
  - New `DeviceExtractor` class
  - New `DeviceParameterCompare` class

Current Work: (1) LVS

- **LVS = Layout Versus Schematic**
- **4 major steps:**
  - Device recognition → `DeviceExtractor::setup`
  - Device parameter calculation → `DeviceExtractor::extract_devices`
  - Connectivity extraction → `connect`
  - Netlist comparison → `DeviceParameterCompare::less`

- **Support of new device types requires**
  - New `DeviceExtractor` class
  - New `DeviceParameterCompare` class

Current Work: (2) Assembly Rule Check

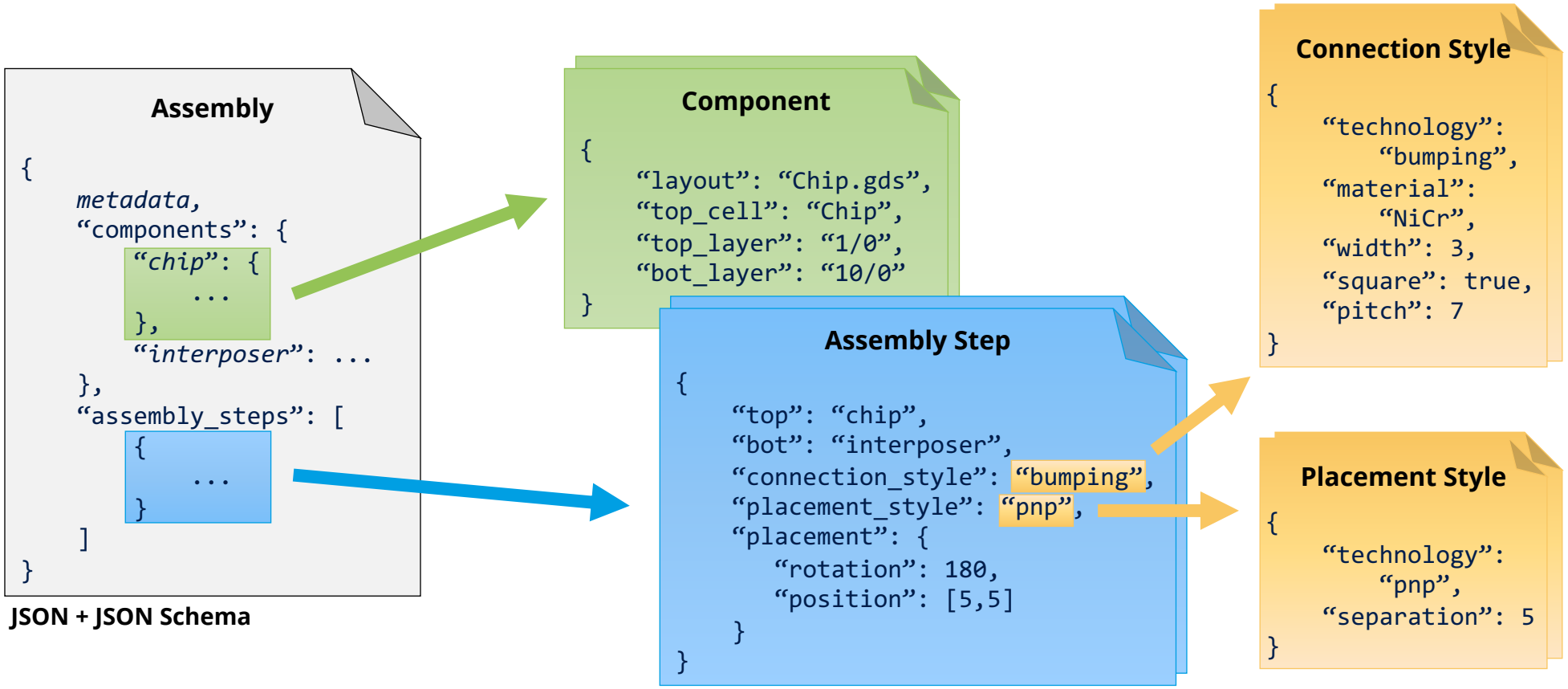


- Advanced packaging requires verification of **assembly rules**
- Sources of assembly rules:
  1. **Assembly technology** for individual dies (e.g. pick and place, micro transfer printing)
  2. **Connection technology** (e.g. wire bonding, bumps, micro bumps)
- Our goals:
  - Formal description of packaging technologies
  - Automatic generation of DRC runsets for KLayout

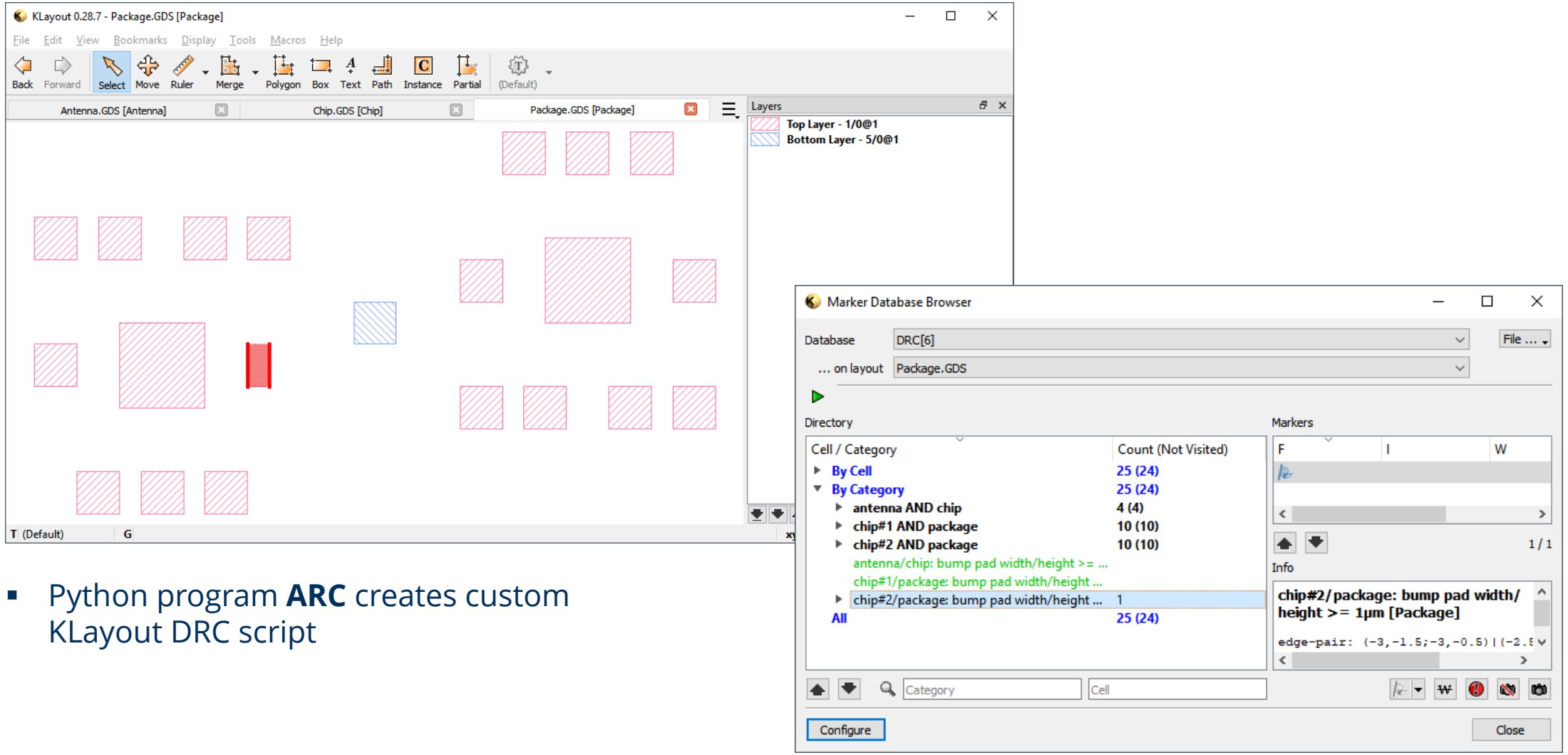
Source: Mohamed Ali *et al.*, „A Versatile SoC/SiP Sensor Interface for Industrial Applications: Implementation Challenges". In: IEEE Access 10 (2022). DOI: 10.1109/ACCESS.2022.3152379

Current Work: (2) Assembly Rule Check

- **Approach:** Extensible Data formats (JSON + JSON Schema) to describe **components**, **assemblies, assembly steps,** as well as **placement & connection styles**



Current Work: (2) Assembly Rule Check



- Python program **ARC** creates custom KLayout DRC script

AG EDA ifte

Open-Source Software for Layout Verification

Tool	DRC	LVS	PEX
KLayout	✓	✓	
Magic VLSI	✓	✓	✓ (*R*, *C*)
Whiteley Research Xic	✓	✓	✓
FastFieldSolvers			✓
OpenRCX			✓ (*R*, *C*)

PEX