Introduction to Silicon Photonics Circuit Design

Wim Bogaerts

Short Course 454 - OFC 2018

GHENT UNIVERSITY

imec embracing a better life

LUCEDA PHOTONICS

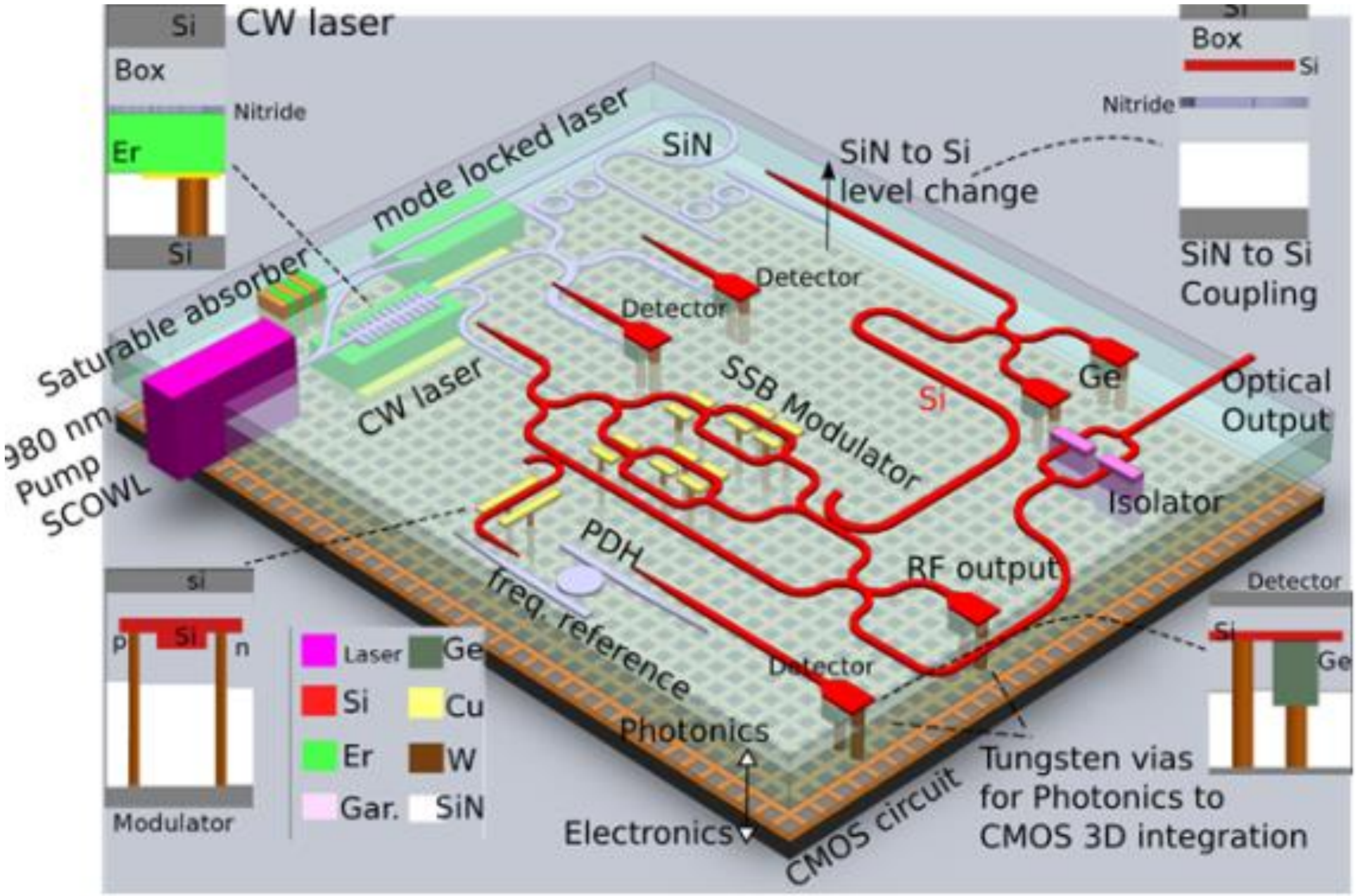
What is silicon photonics?

The implementation of high density photonic integrated circuits by means of CMOS process technology in a CMOS fab

Enabling complex optical functionality on a compact chip at low cost

PHOTONIC INTEGRATED CIRCUITS (PIC)

Integration of (many) optical functions on a chip



Source: EECS Berkeley

INDUSTRIAL TAKE-UP EXAMPLES IN TELECOM/DATACOM/DATA CENTERS

- active optical cables (eg PSM4: 4x28 Gb/s on parallel fibers)
- WDM transceivers (eg 4 WDM channels x 25 Gb/s on single fiber)
- coherent receiver (eg 100 Gb/s PM-QPSK)
- fiber-to-the-home bidirectional transceiver (eg 12 x 2.5 Gb/s)
- monolithic receiver (eg 16x20Gb/s)
- 40Gb/s, 50Gb/s and 100 Gb/s Ethernet (future: 400Gb/s)
- …

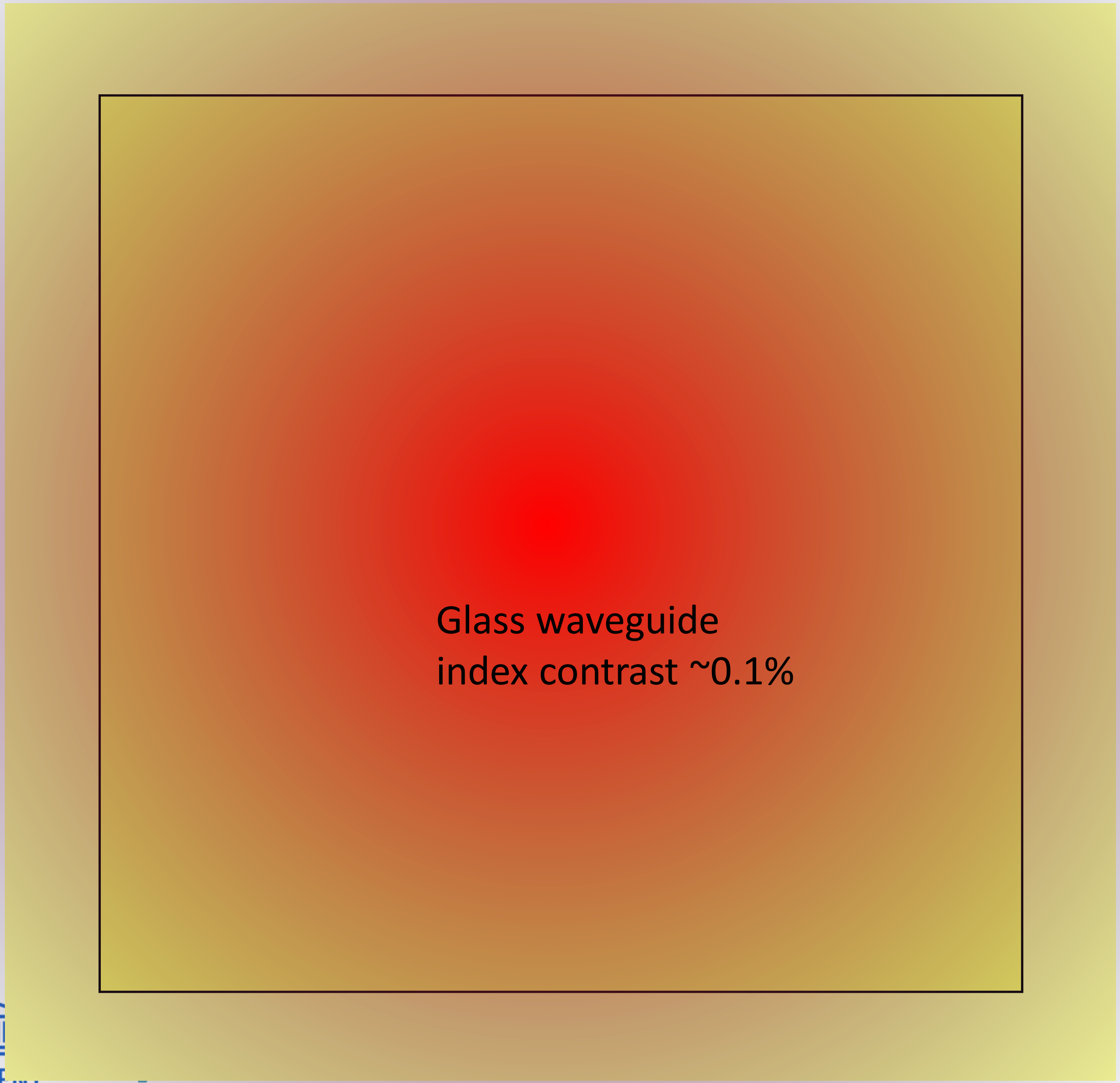
WHY SILICON PHOTONICS?

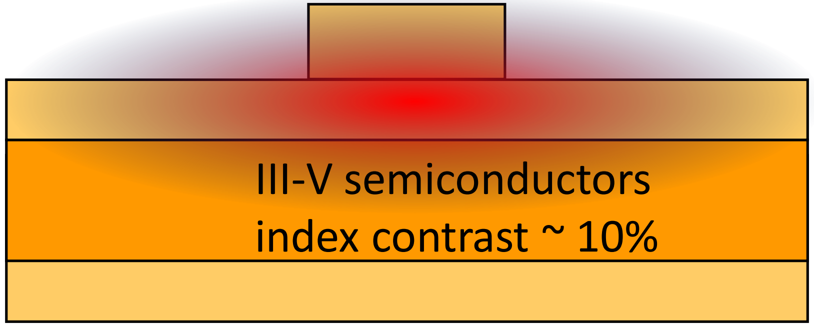
Large scale manufacturing

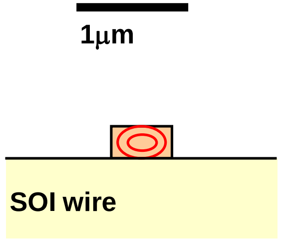
Scale

Submicron-scale waveguides

Scaling optical waveguides: Index contrast





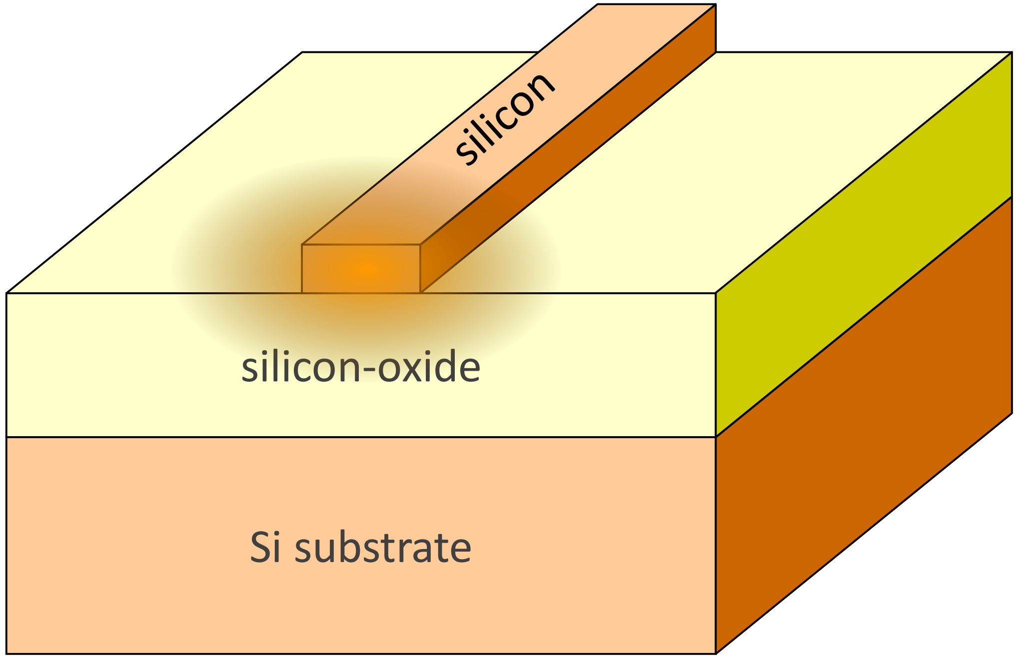


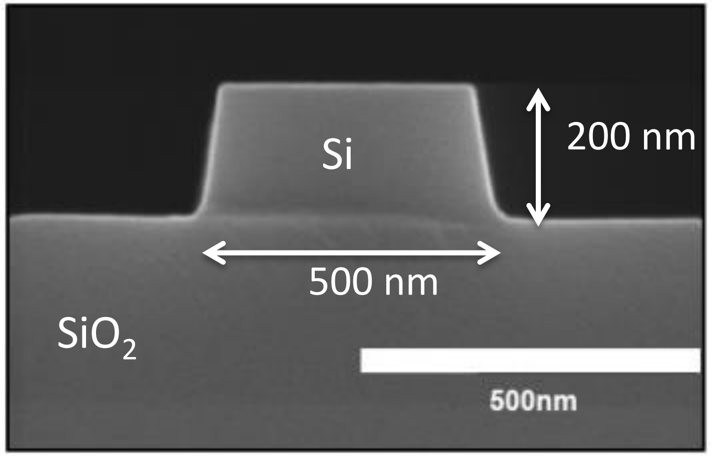
silicon wire:
index contrast ~ 200%

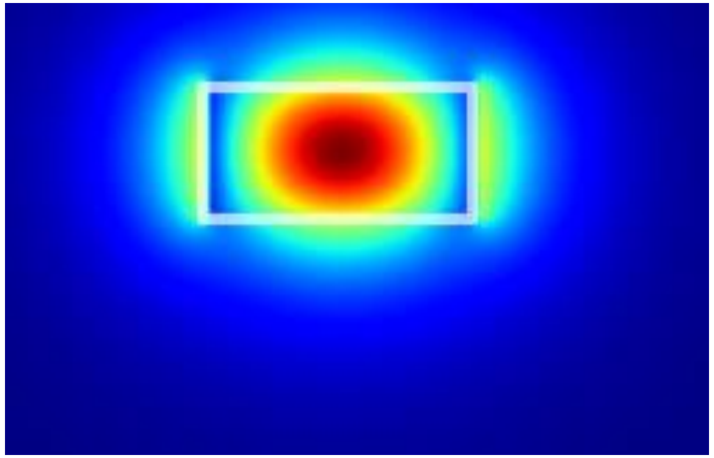
Waveguides: Silicon photonic wires

High contrast waveguides

- submicrometer dimensions
- small bend radius

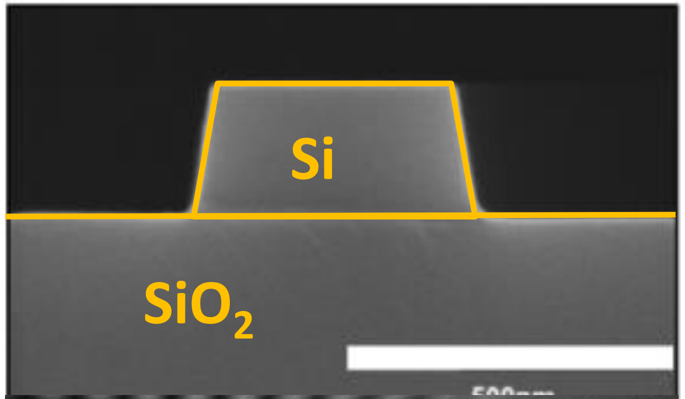




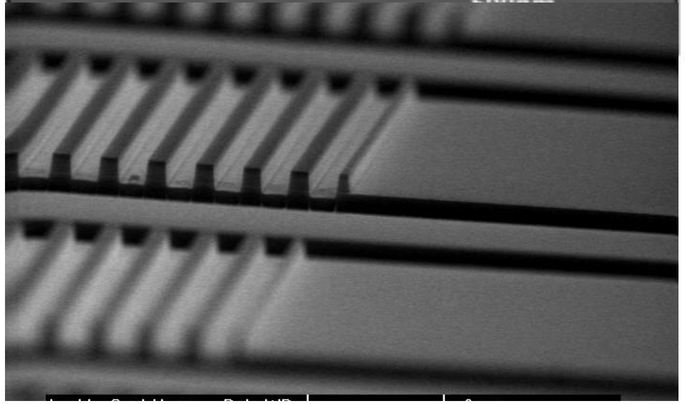


optical mode

HIGH INDEX CONTRAST: A BLESSING AND A CURSE

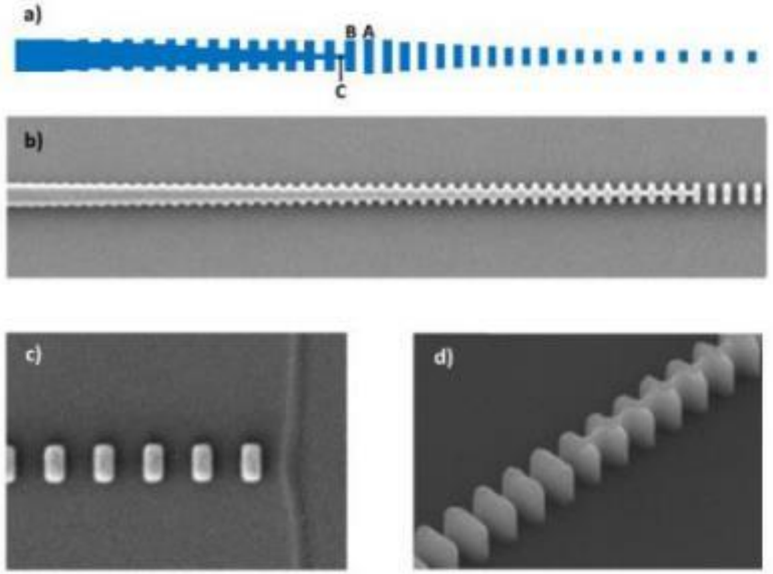


Very tight confinement of light



Very small bend radii : down to 1 μm

Very dense integration of components on a chip



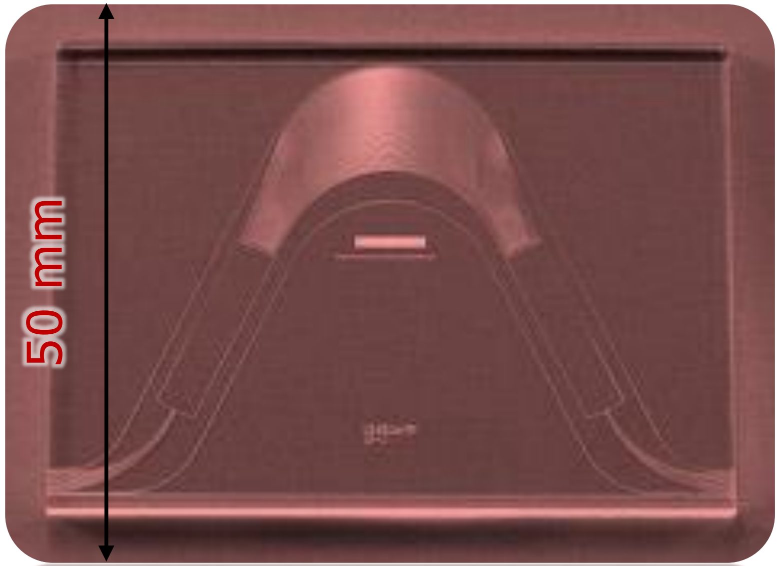
Sub-wavelength design freedom

Photonic crystals with extremely high quality cavities

…

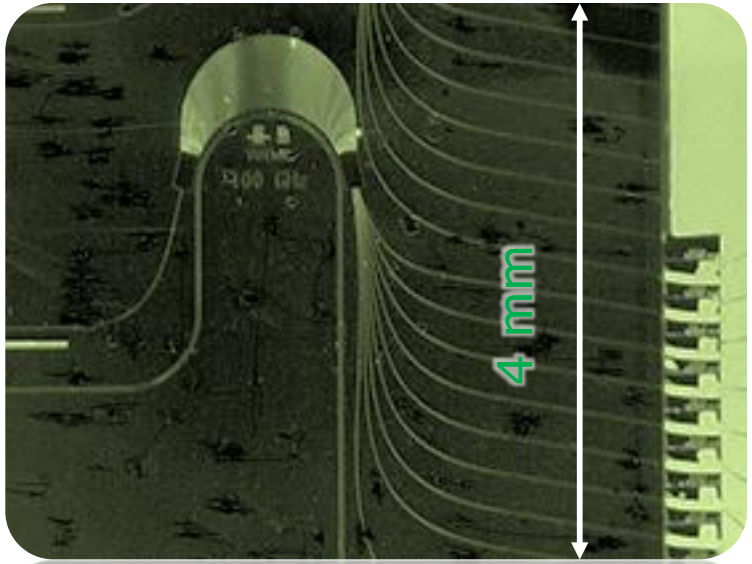
Cheben, OE, 2015

Higher contrast, smaller cores, tighter bends



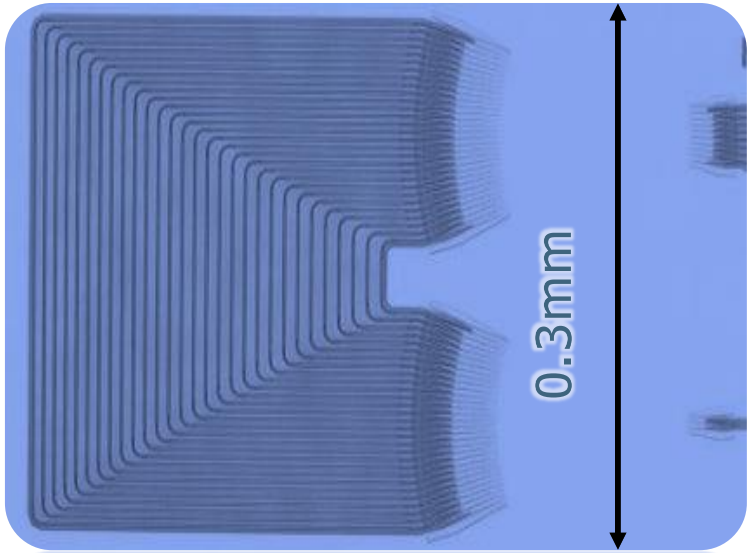
Silica on silicon

Contrast ~ 0.01 – 0.1
Mode diameter ~ 8µm
Bend radius ~ 5mm
Size ~ 10 cm^2



Indium Phosphide

Contrast ~ 0.2 – 0.5
Mode diameter ~ 2µm
Bend radius ~ 0.5mm
Size ~ 10mm^2



Silicon on insulator

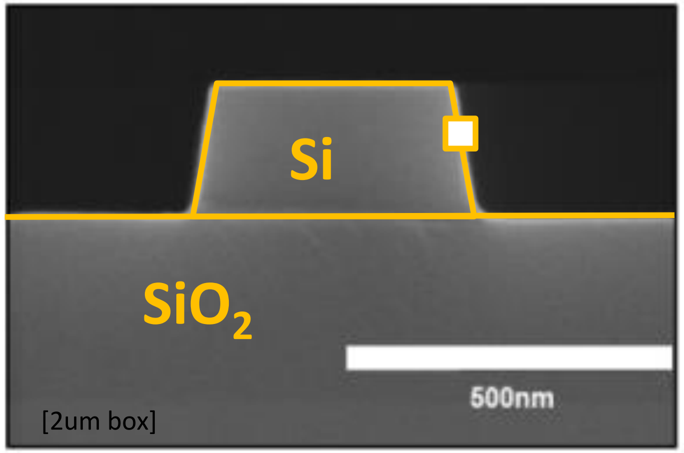
Contrast ~ 1.0 – 2.5
Mode diameter ~ 0.4µm
Bend radius ~ 5µm
Size ~ 0.1mm^2

10000 ×

HIGH INDEX CONTRAST: A BLESSING AND A CURSE

Every nm^3 matters

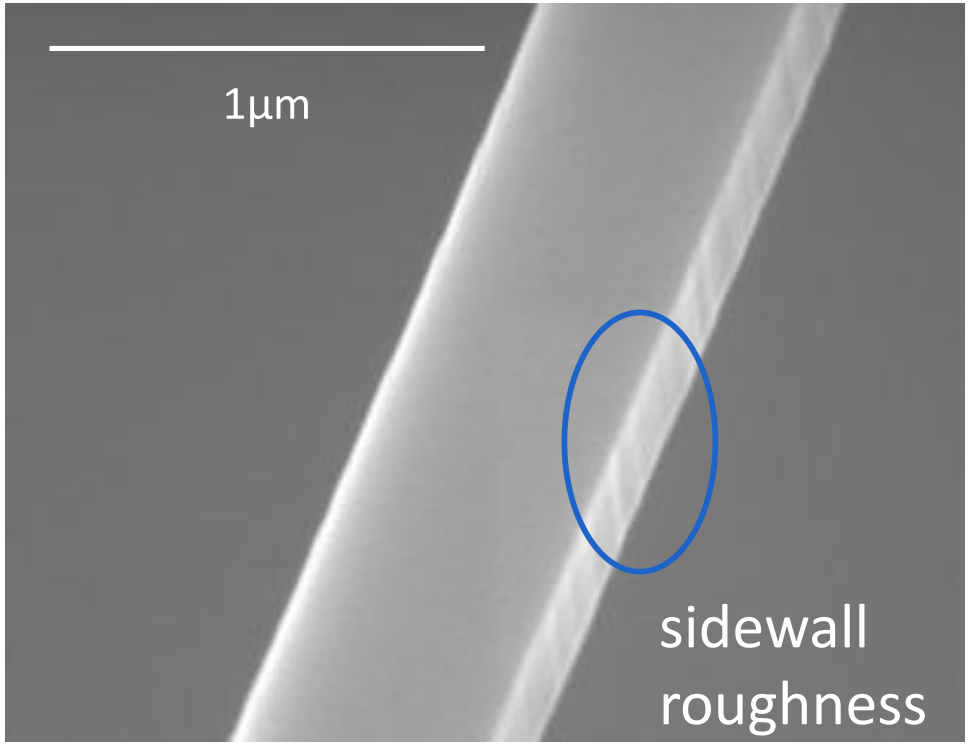
CMOS technology is the only manufacturing technology with sufficient nm-process control to take advantage of the blessing without suffering from the curse

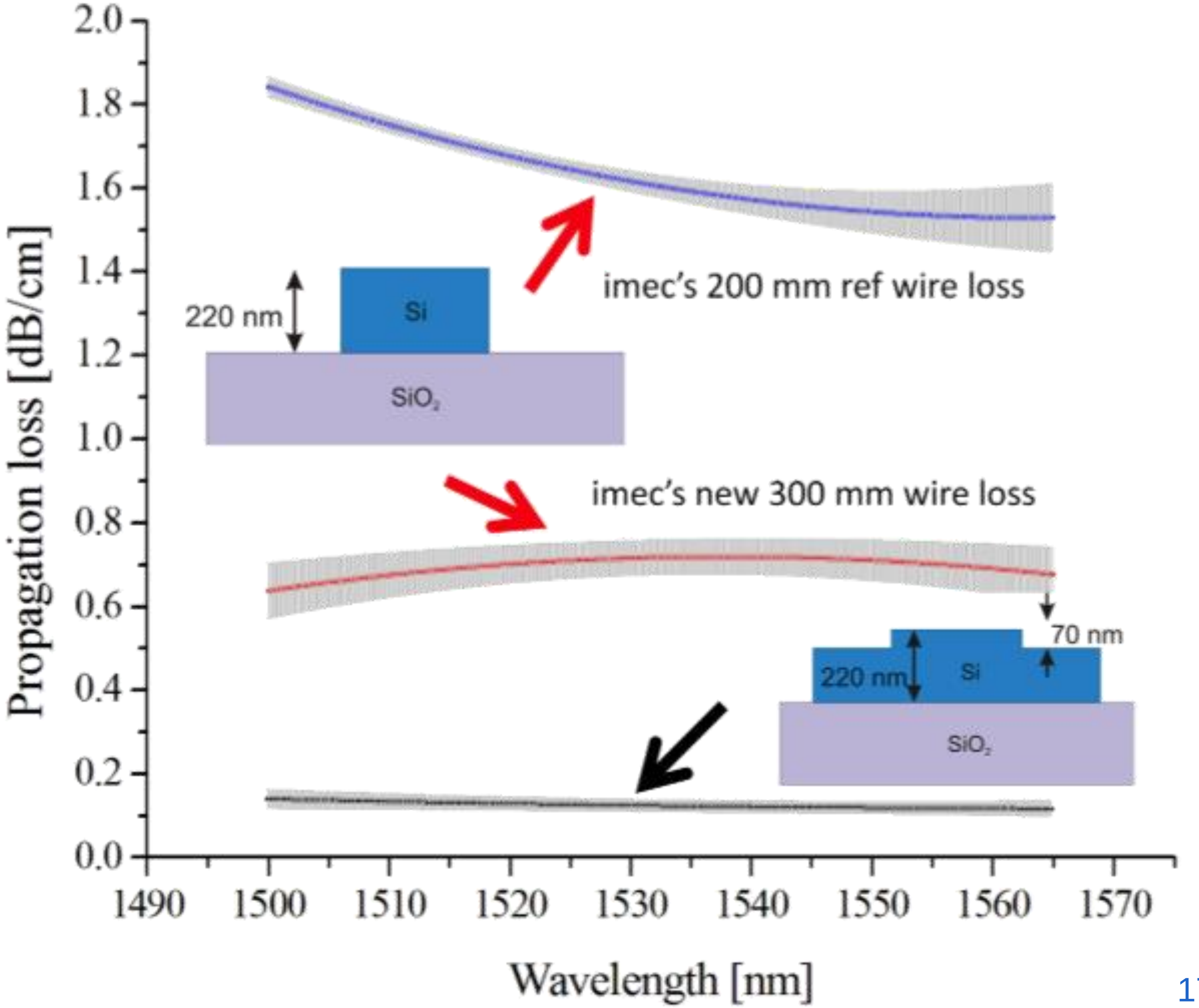


WAVEGUIDES

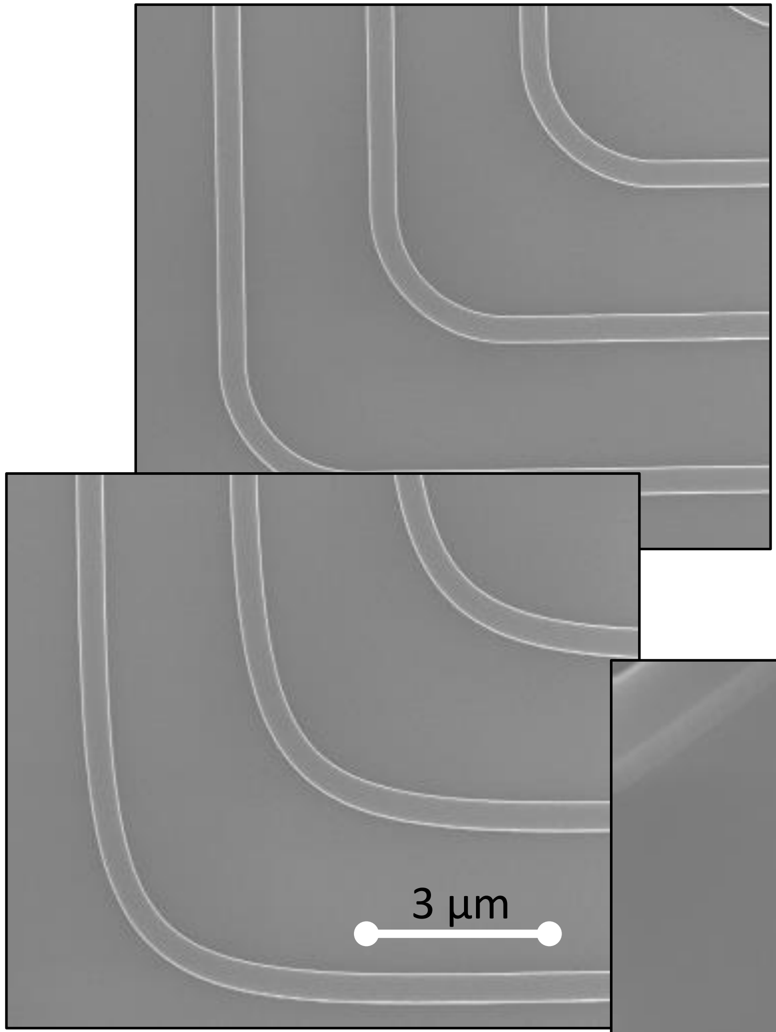
Waveguide losses dominated by scattering.

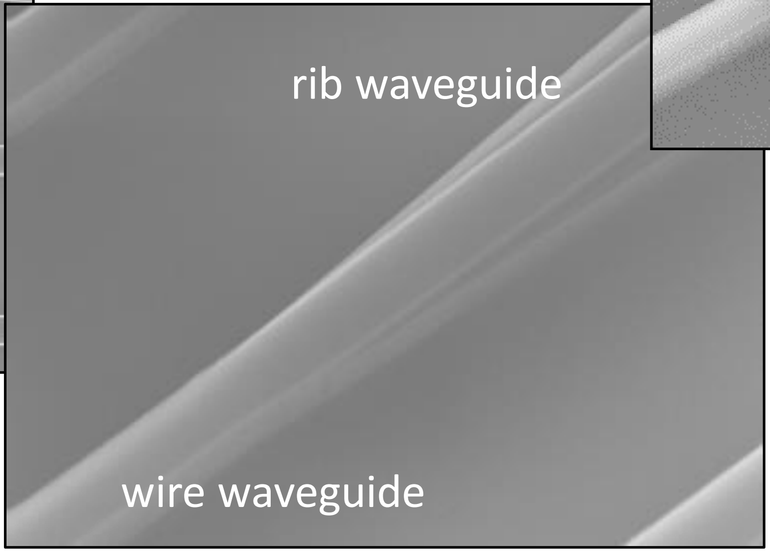
Use better litho + etch

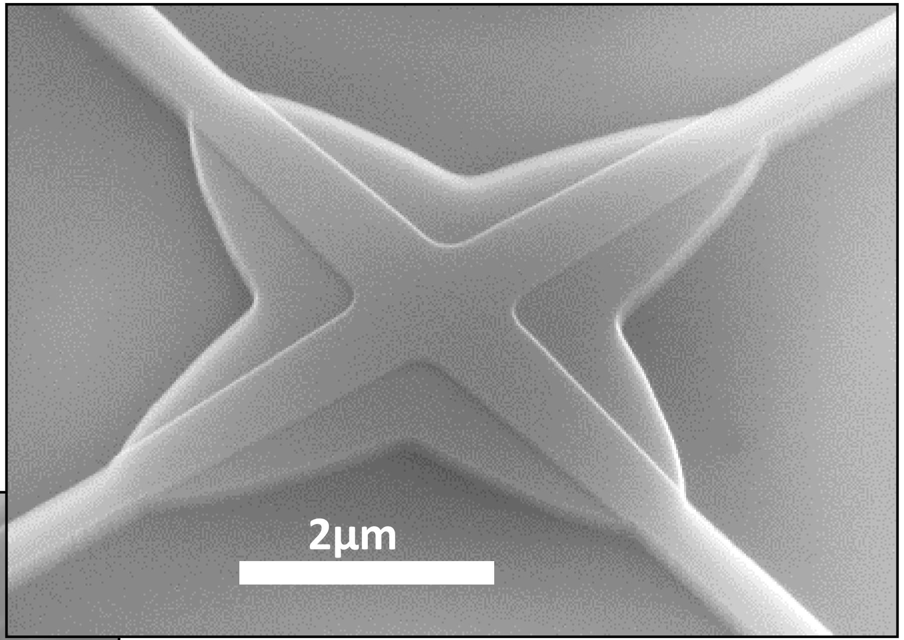




COMPACT BENDS, TRANSITIONS, CROSSINGS



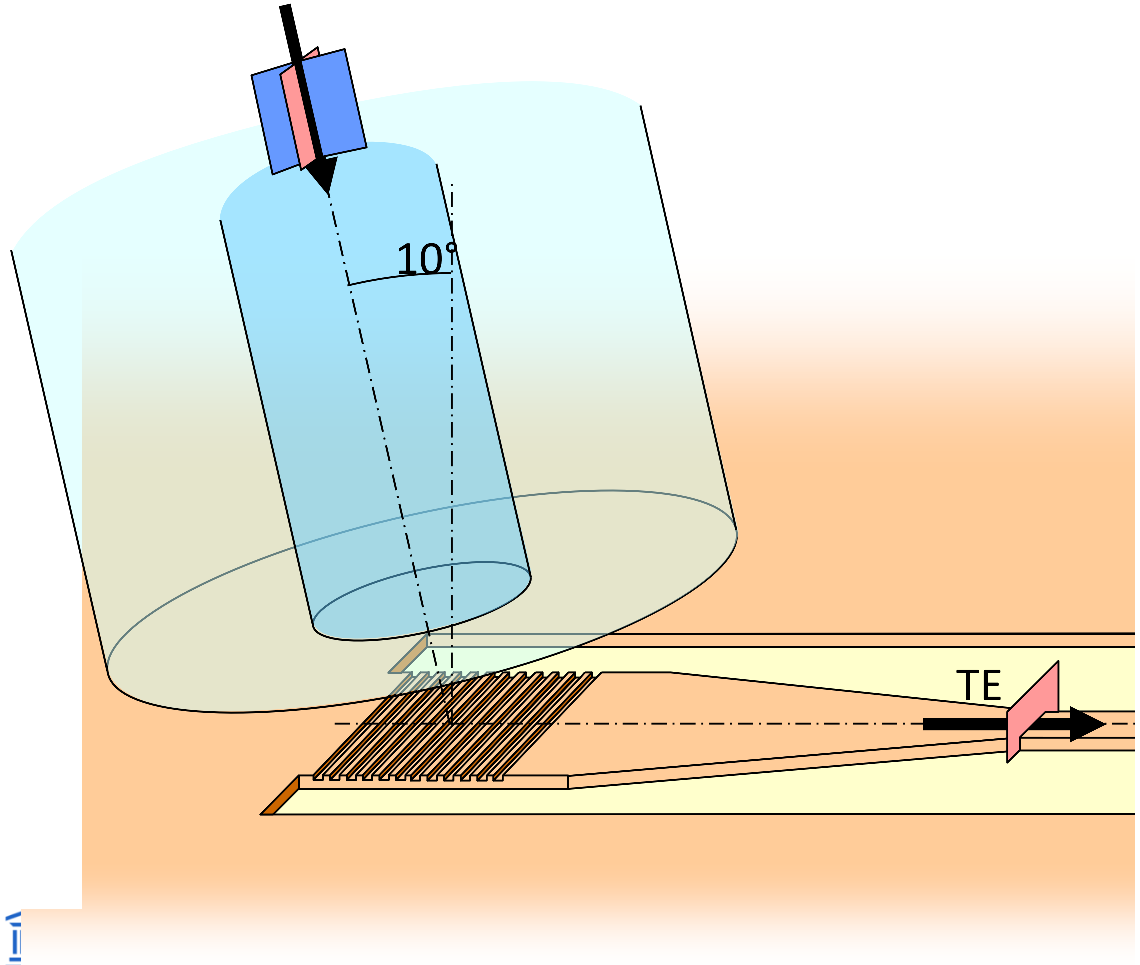


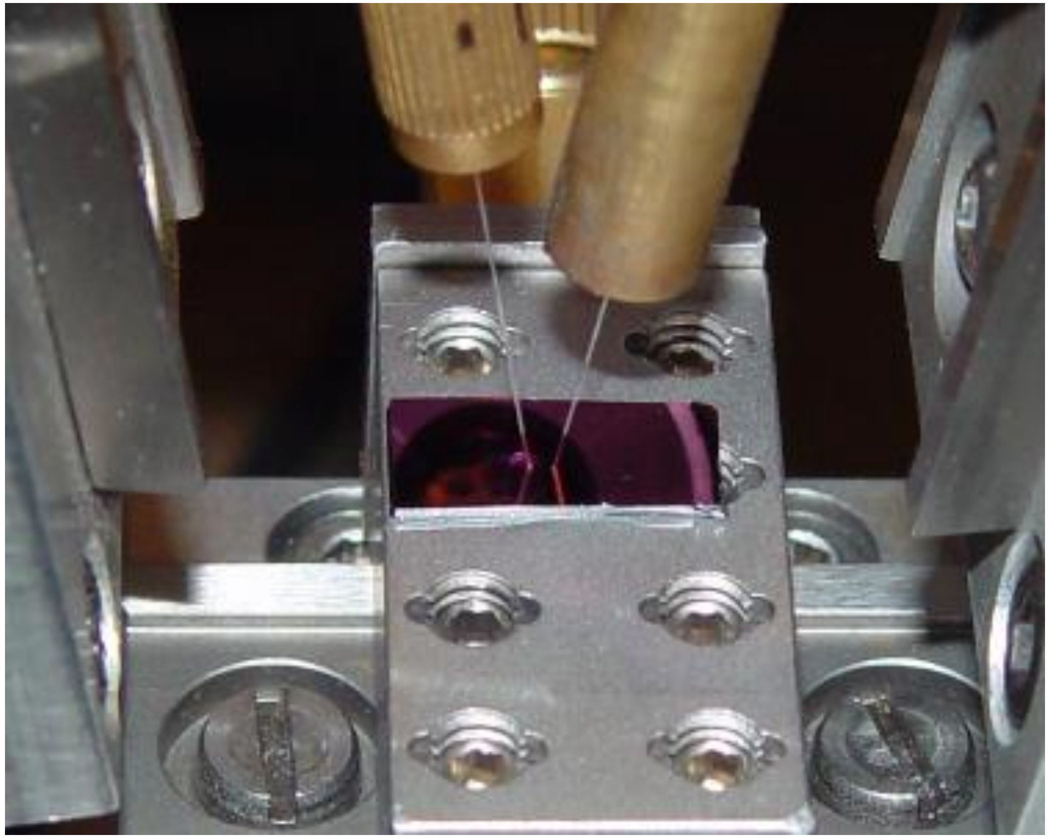


Waveguide Crossing: 97% transmission

FIBER-TO-CHIP COUPLING

Vertical fiber interface: allows easy on-chip testing



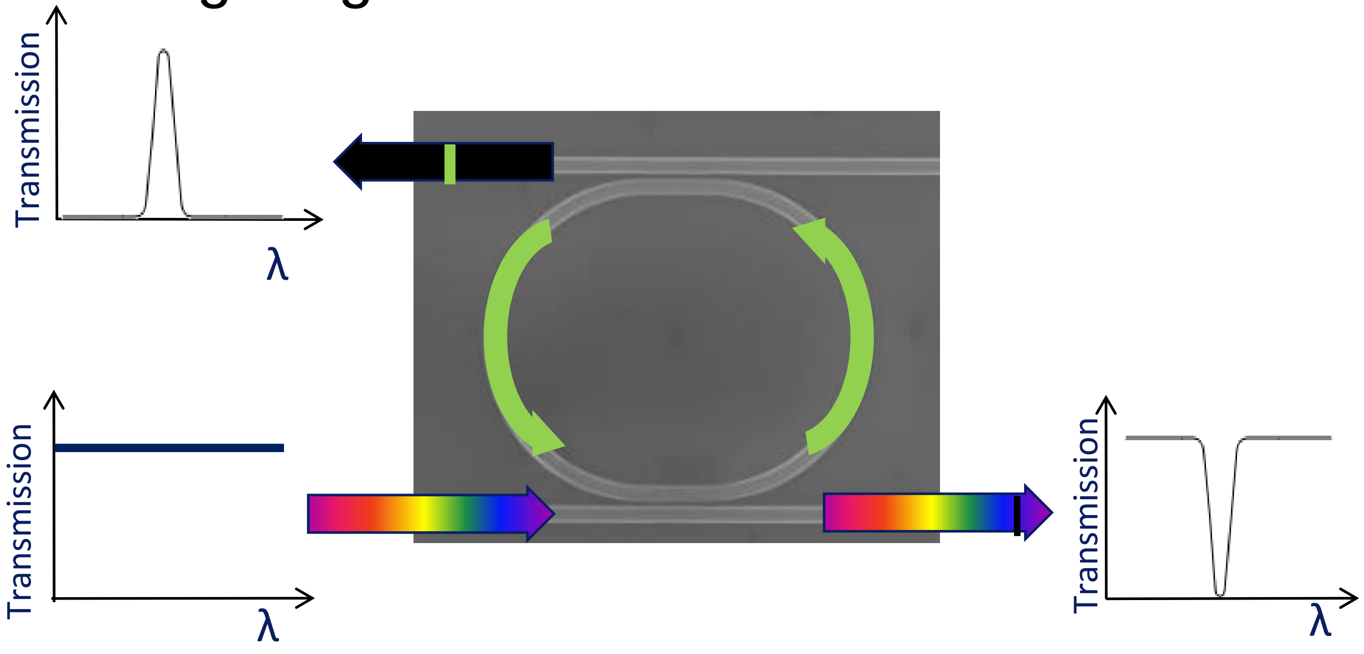


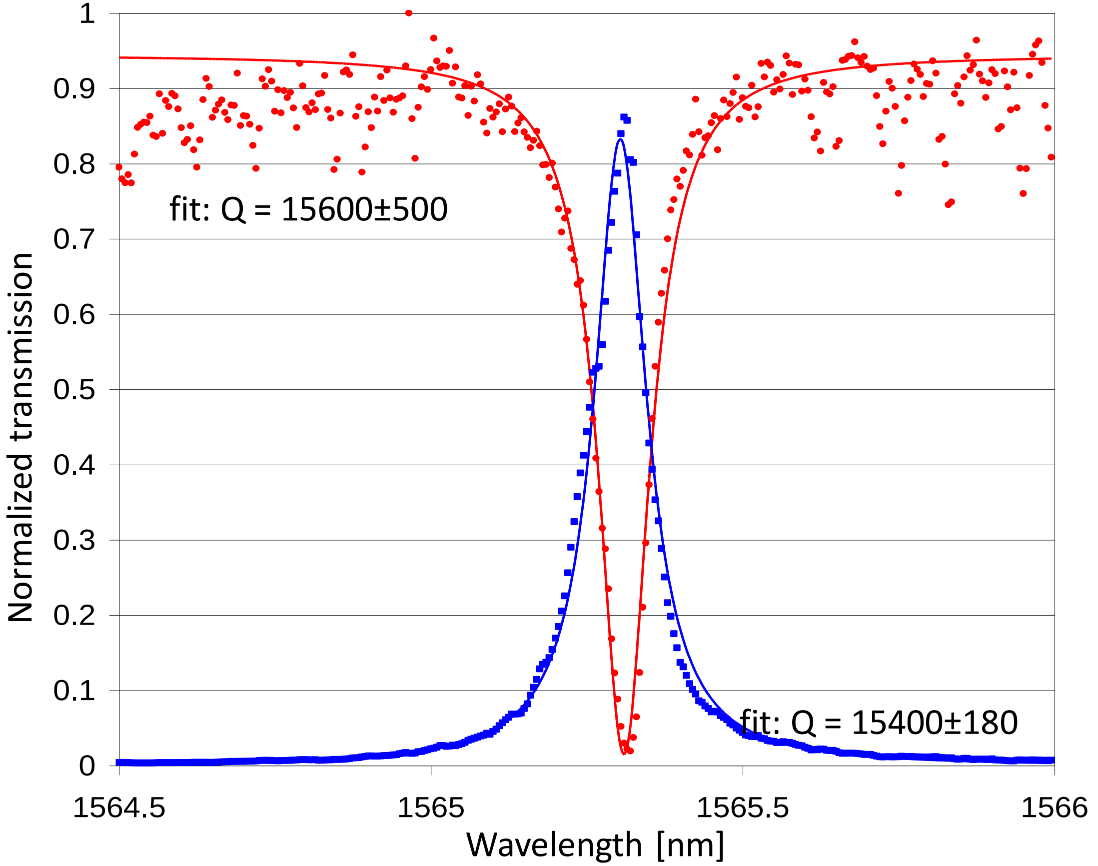
Wavelength Filtering Functions

Light is a wave: interference at the 100nm scale

- interferometers
- resonators

e.g. ring resonator

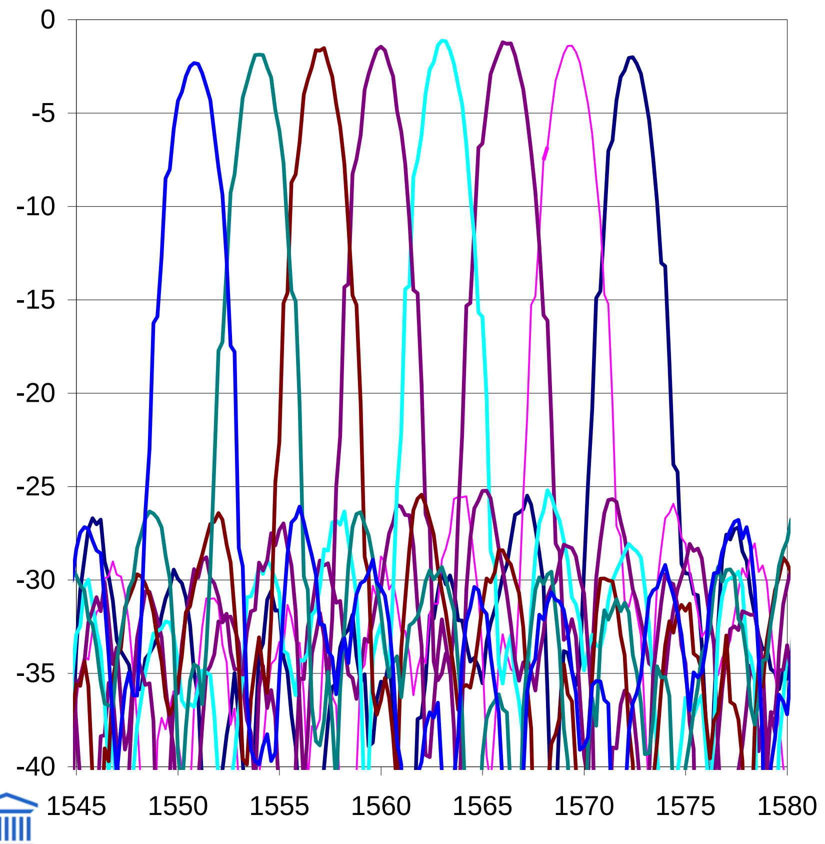


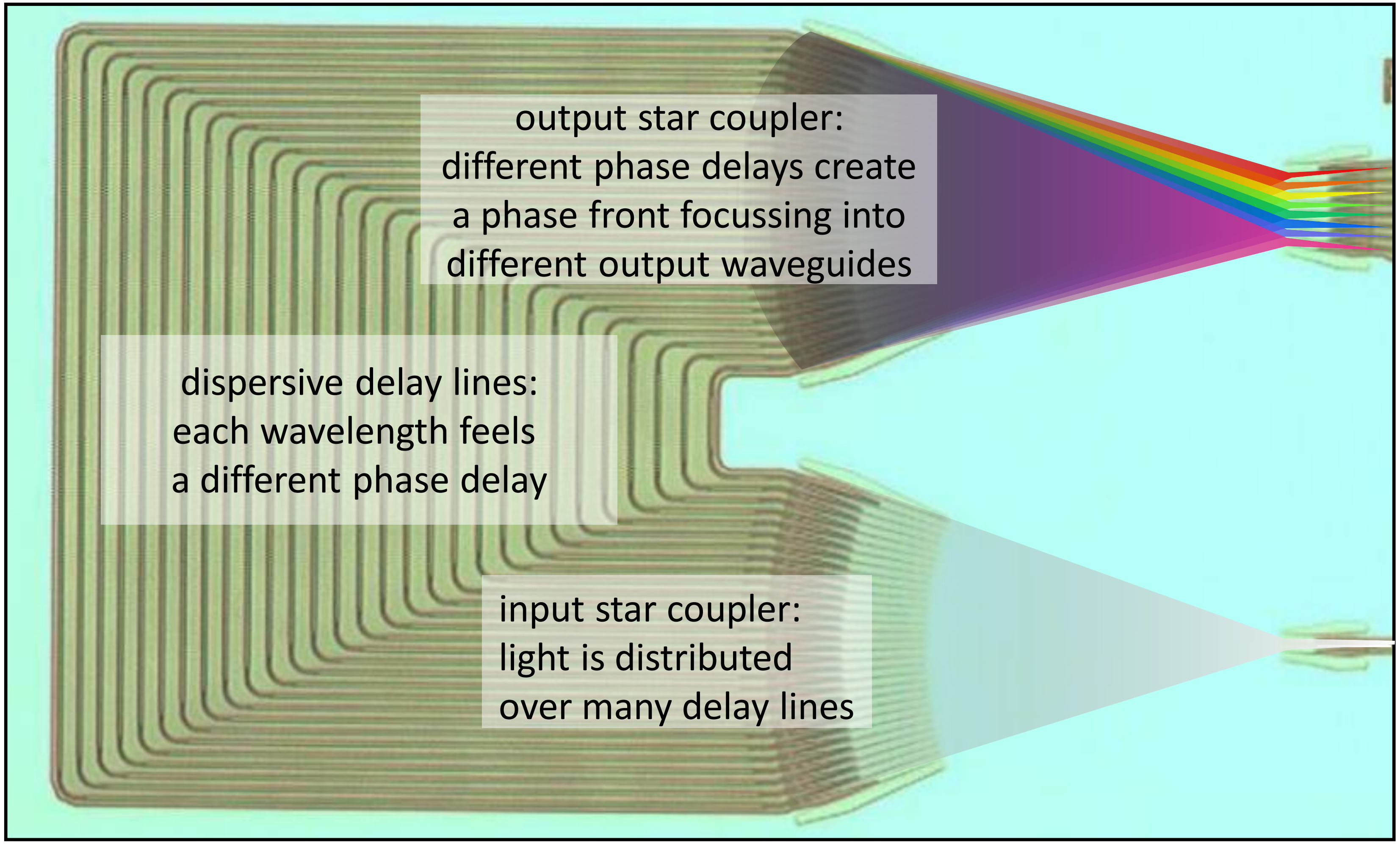


GHENT UNIVERSITY imec

Wavelength filtering functions

Arrayed waveguide grating





GHENT UNIVERSITY imec

SENSITIVITY OF SILICON PHOTONICS WAVELENGTH FILTERS

Silicon photonic waveguides are sensitive to

- geometry
- stress
- temperature
- …

wire width

$$\frac{\partial\lambda}{\partial w} \approx 1\,{}^{nm}\!/_{nm}$$

wire height

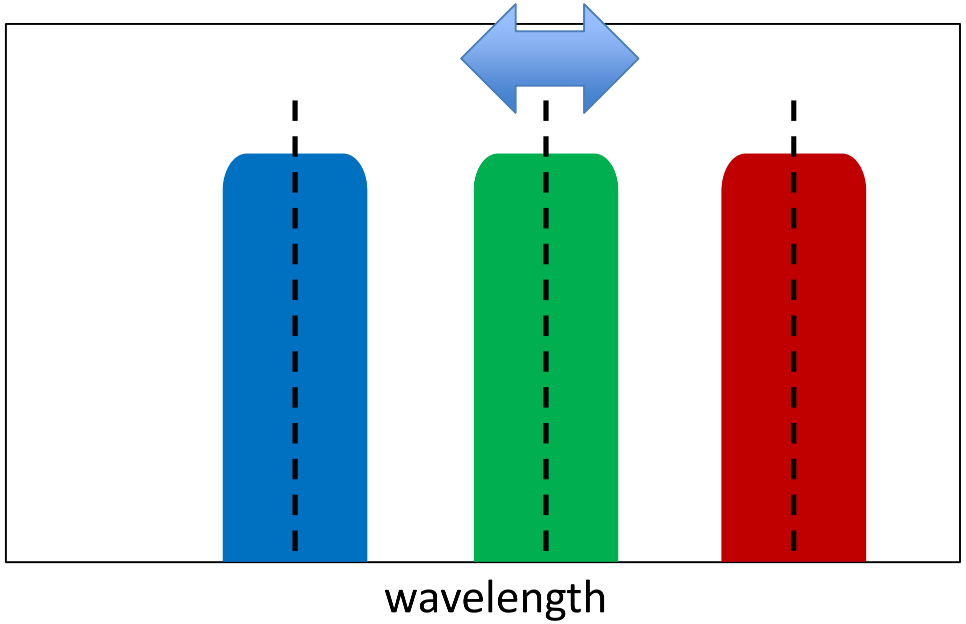
$$\frac{\partial\lambda}{\partial h} \approx 2\,{}^{nm}\!/_{nm}$$

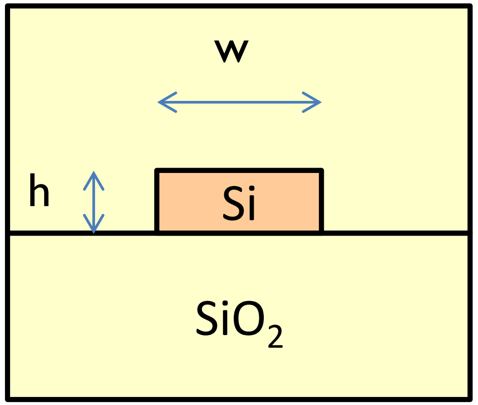
temperature

$$\frac{\partial\lambda}{\partial T} \approx 0.08\,{}^{nm}\!/_{K}$$

1nm ~20K

1nm ~10K



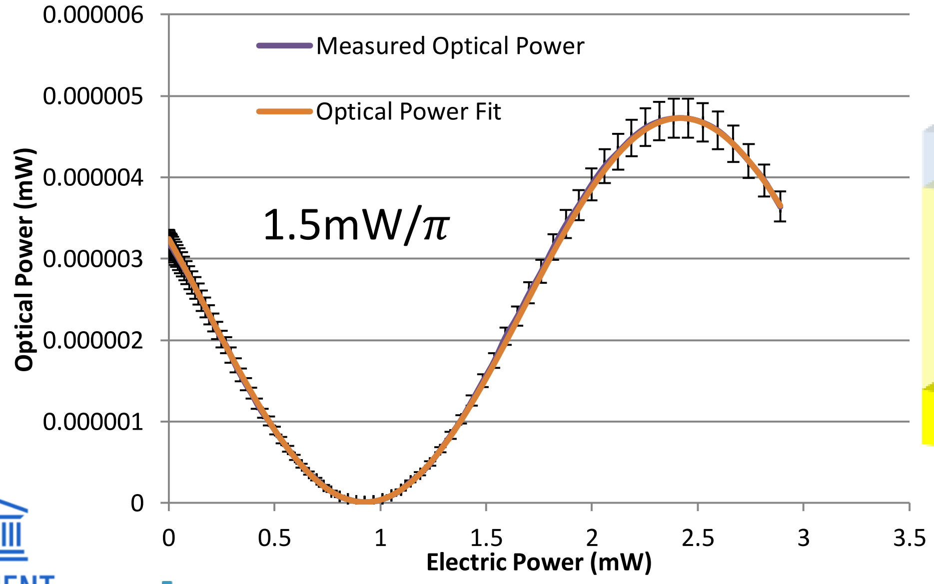


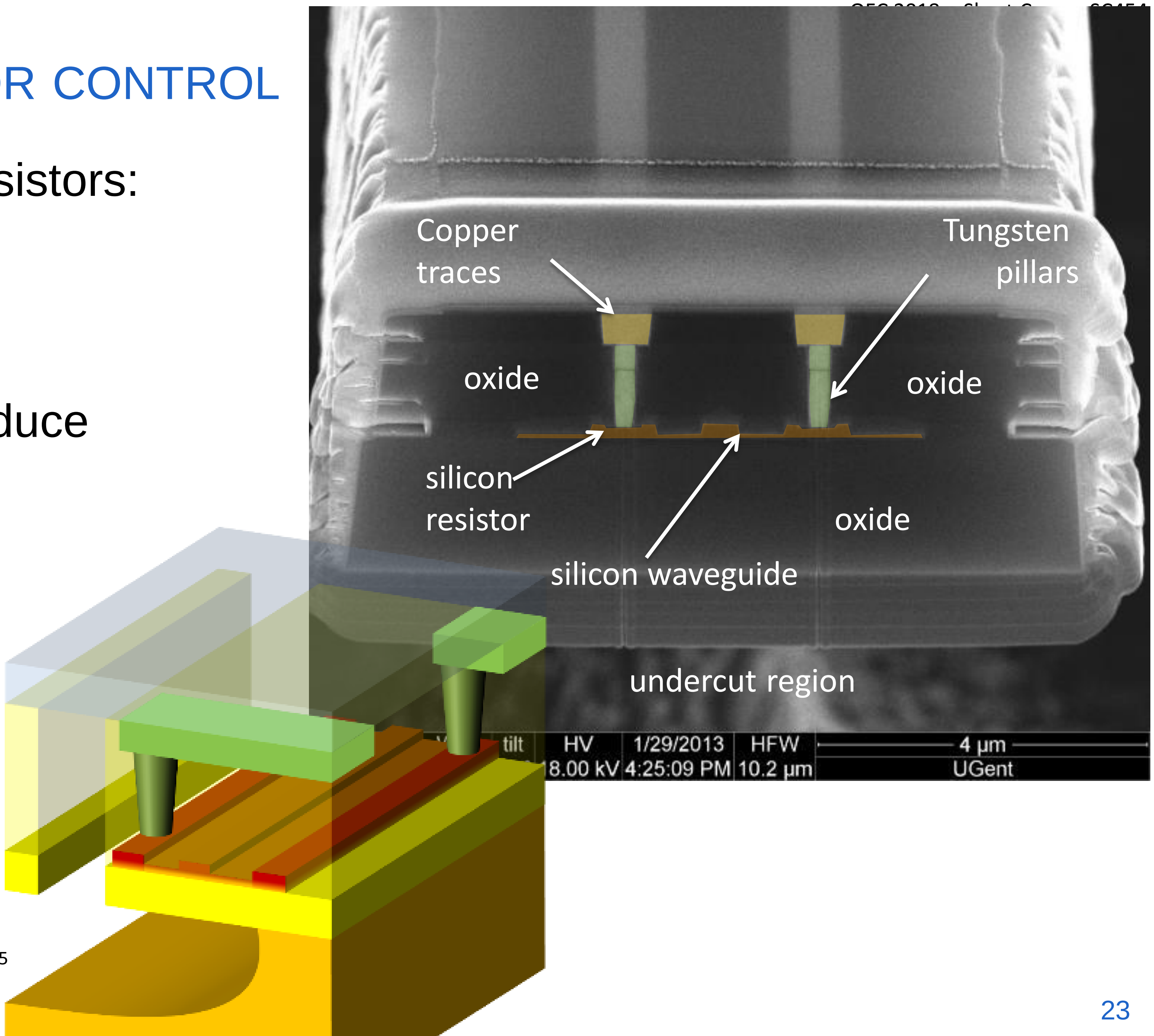
Integrated heaters for control

Different types of electrical resistors:

metal, silicide, doped silicon

Optional undercut to lower reduce thermal leakage.



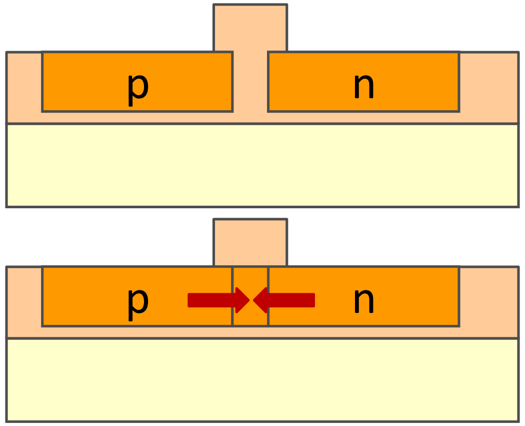


GHENT UNIVERSITY imec

ELECTRO-OPTIC EFFECT IN SILICON: INJECTION VS. DEPLETION

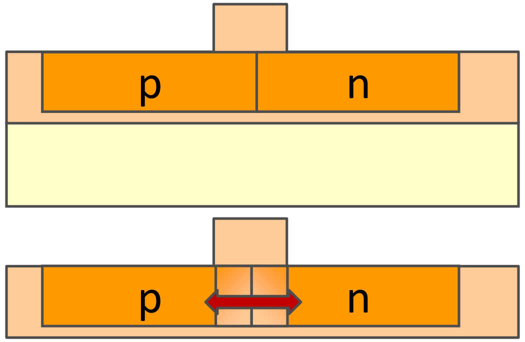
Carrier injection

- p-i-n diode in forward bias
- Inject carriers into waveguides
- Strong effect (many carriers)
- Slow effect (~1GHz)



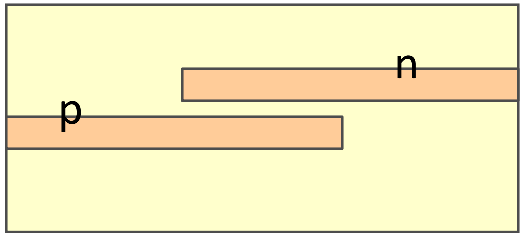
Carrier depletion

- p-n diode in reverse bias
- Extract carriers from waveguide
- Weaker effect
- Fast effect (>40GHz)



Carrier accumulation

- Accumulation at oxide
- Similar to capacitor
- Fast

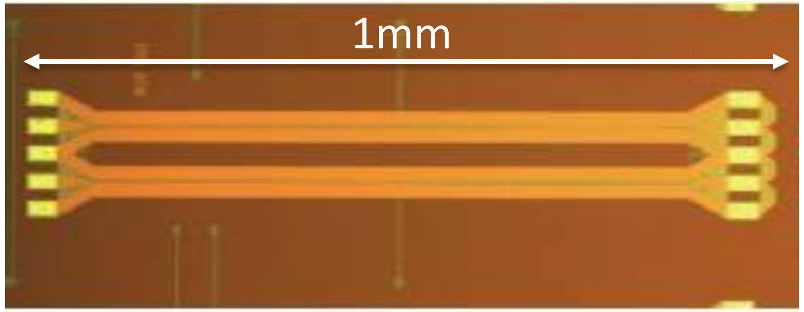


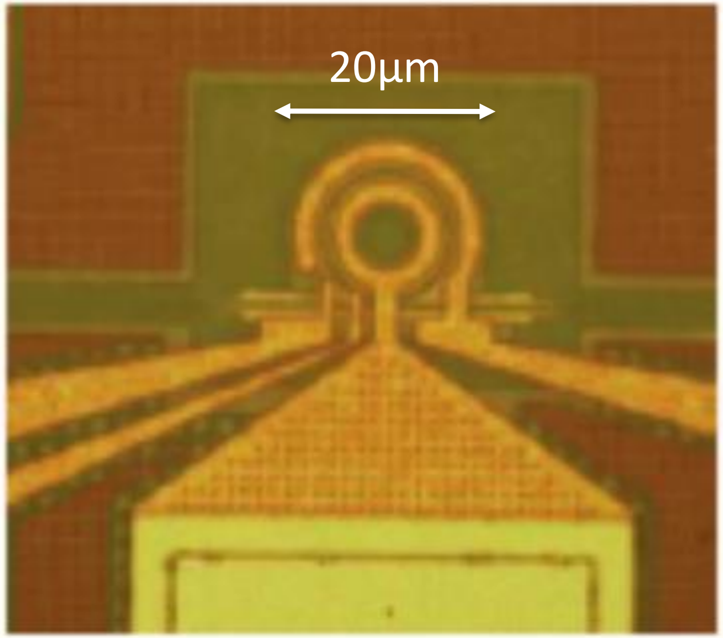
ELECTRICAL SIGNAL MODULATION

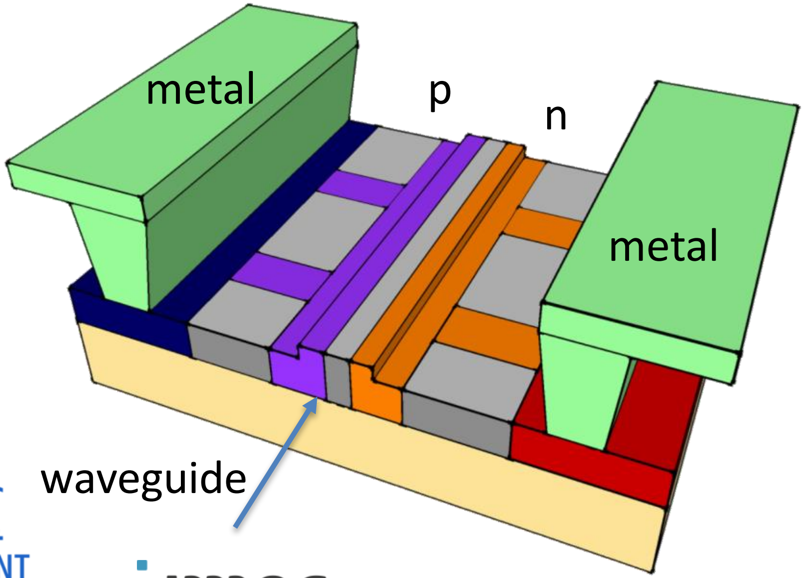
Add doped junction to silicon waveguide:

modulate refractive index

- travelling wave modulator
- ring resonator modulator

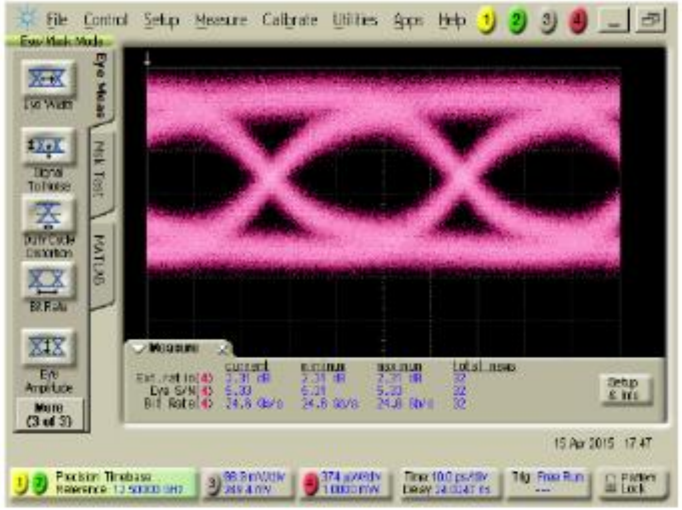






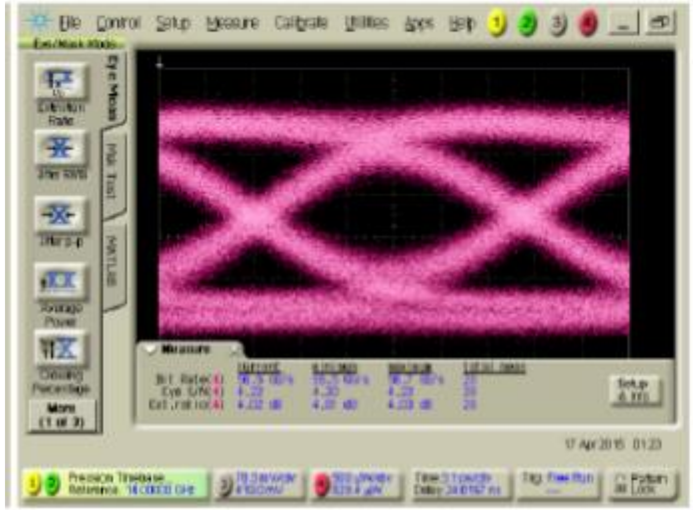
25Gb/s, 1Vpp

Vbias= -0.2V, ER = 2.3dB, Q = 5.3, Opt. Power=13dbm, 1560nm, PRBS=2e31-1



56Gb/s, 2.5Vpp

Vbias=-0.75V, ER=4dB, Q=4.2, PRBS=2e31-1



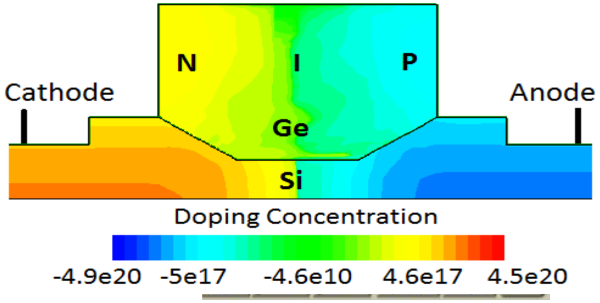
Germanium Electro-Absorption Modulator

Advantages

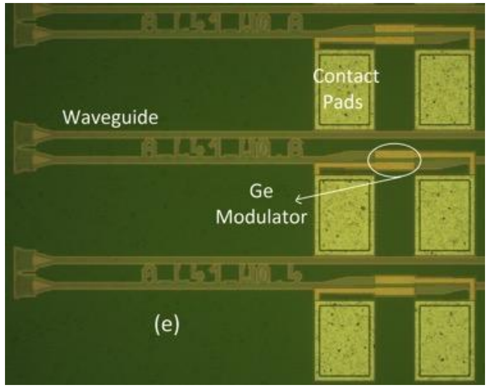
- Uses existing Ge-detector technology
- Compact, optical BW > 35nm
- 3dB BW > 40GHz

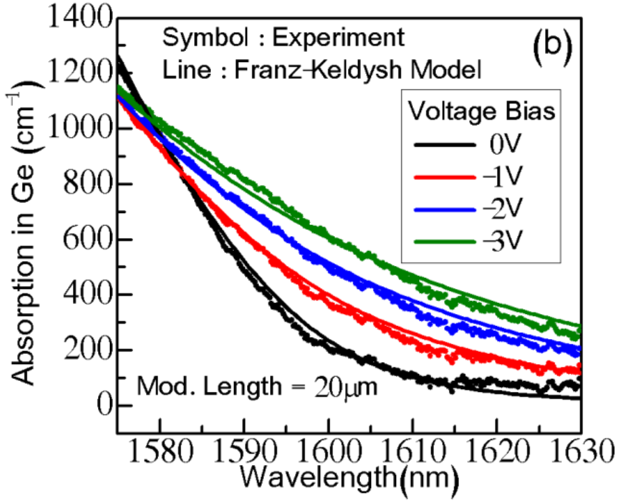
Next steps

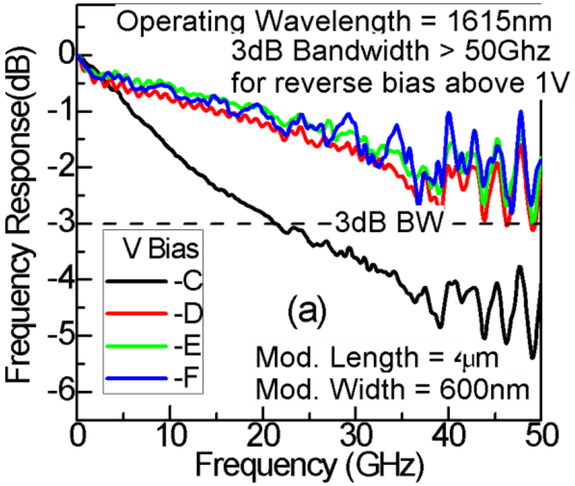
- Ge → SiGe to reach C-band
- Ge-laser



Gupta, S., et al. 50GHz Ge Waveguide Electro-Absorption Modulator Integrated in a 220nm SOI Photonics Platform. In *Optical Fiber Communication Conference* (Vol. 1, pp. 5–7) 2015.



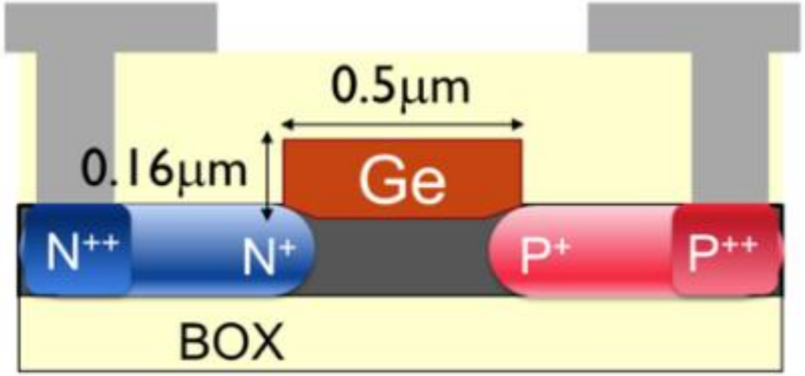


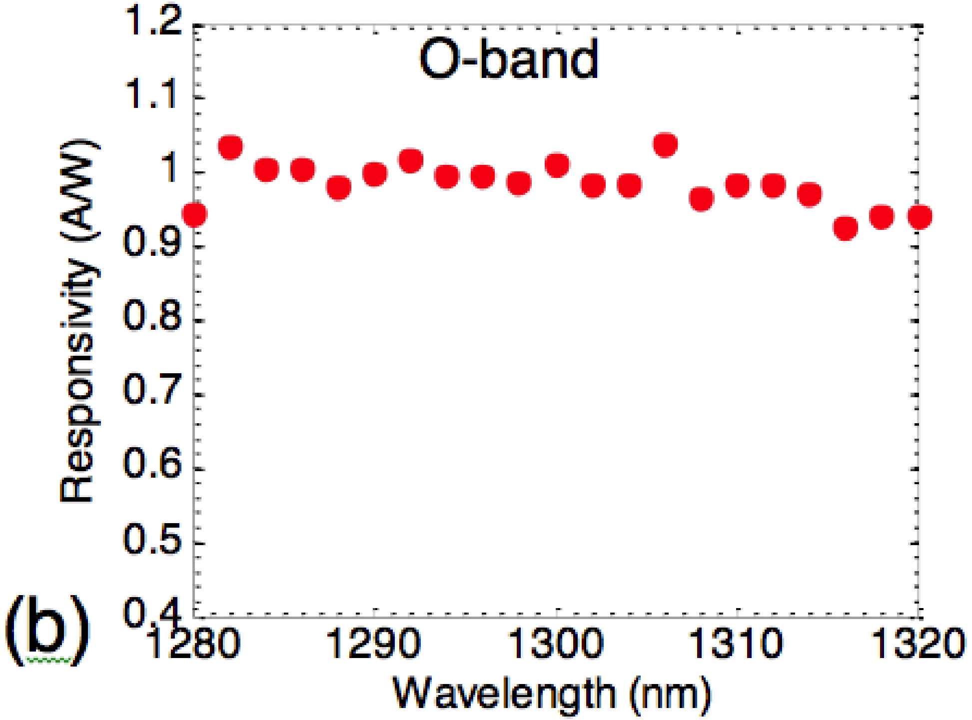


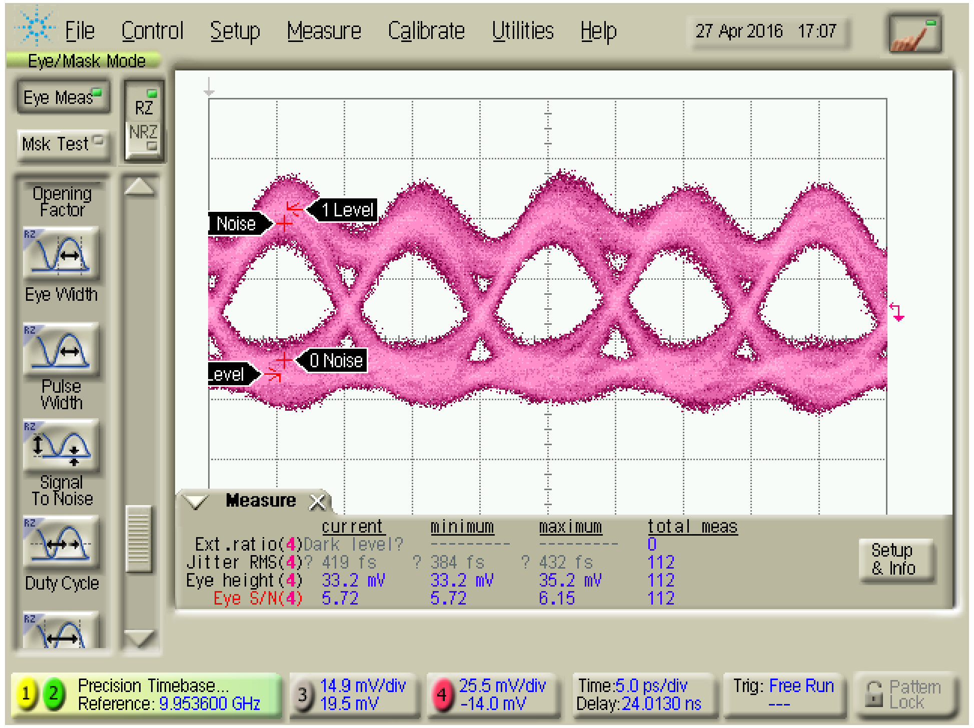
FAST AND EFFICIENT PHOTODETECTORS

Integrated Germanium Photodetectors

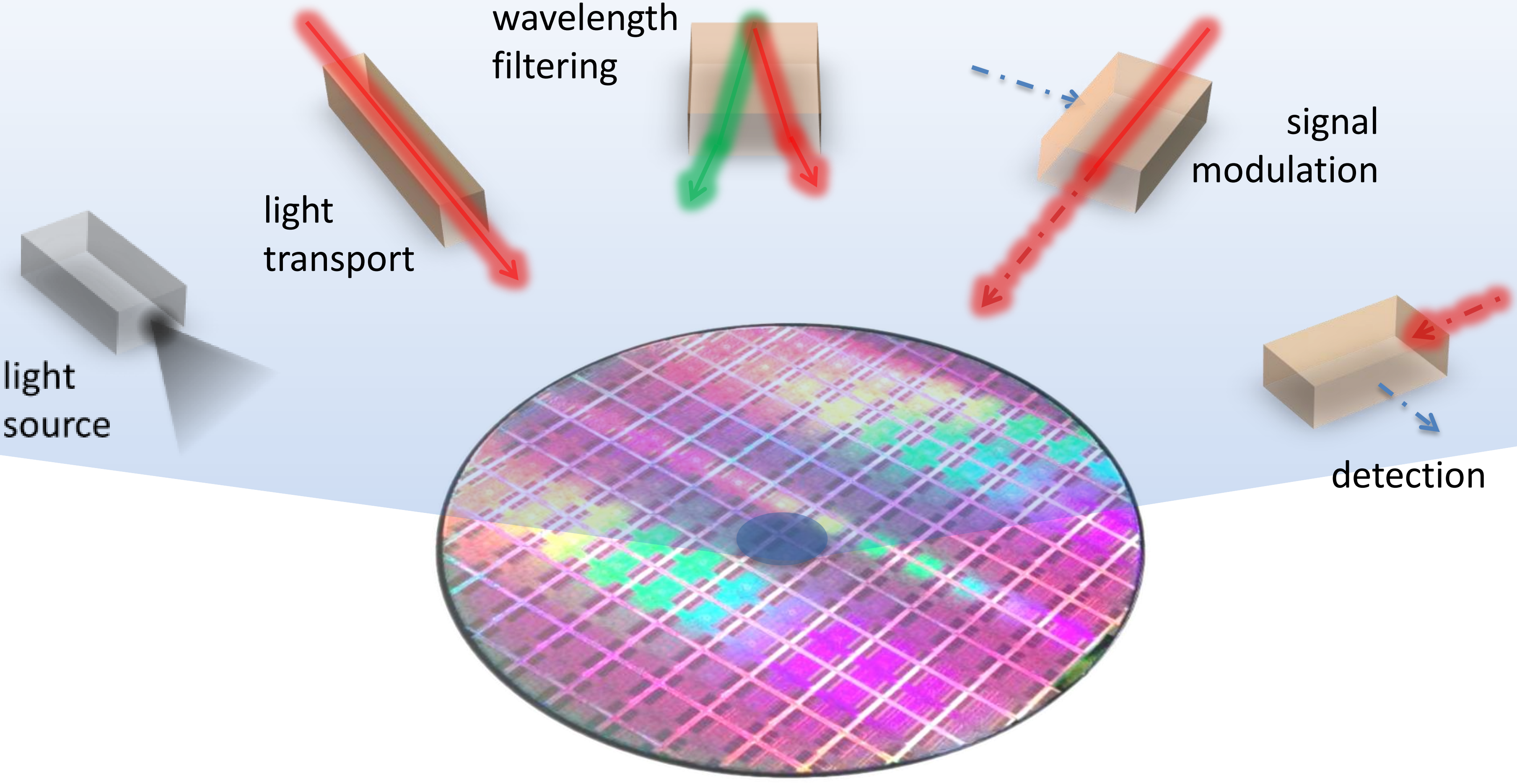
- 1 A/W responsivity
- > 70 GHz bandwidth
- 3nA dark current
- 1 V operation





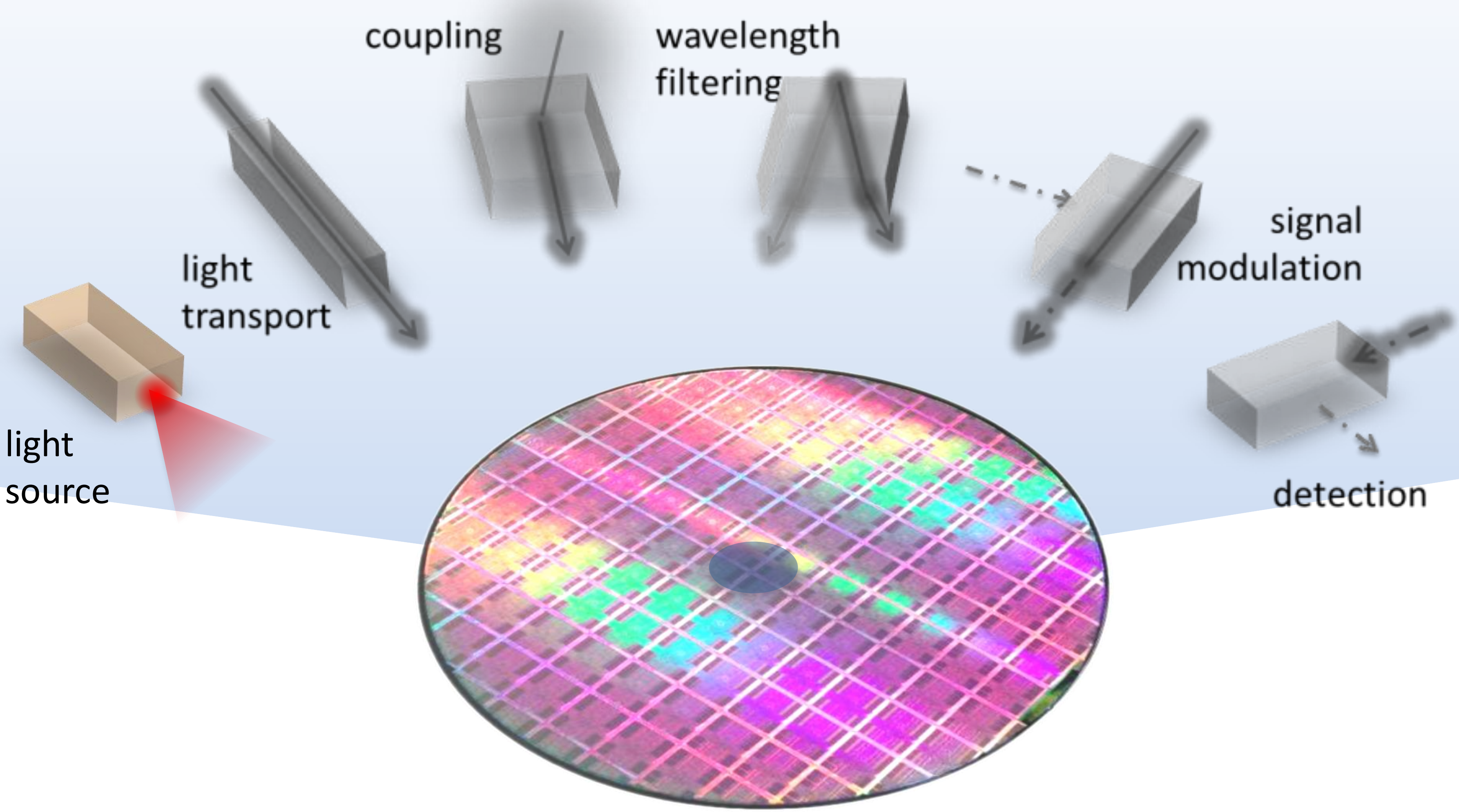


INTEGRATION ON WAFER SCALE



Compatible with CMOS processing

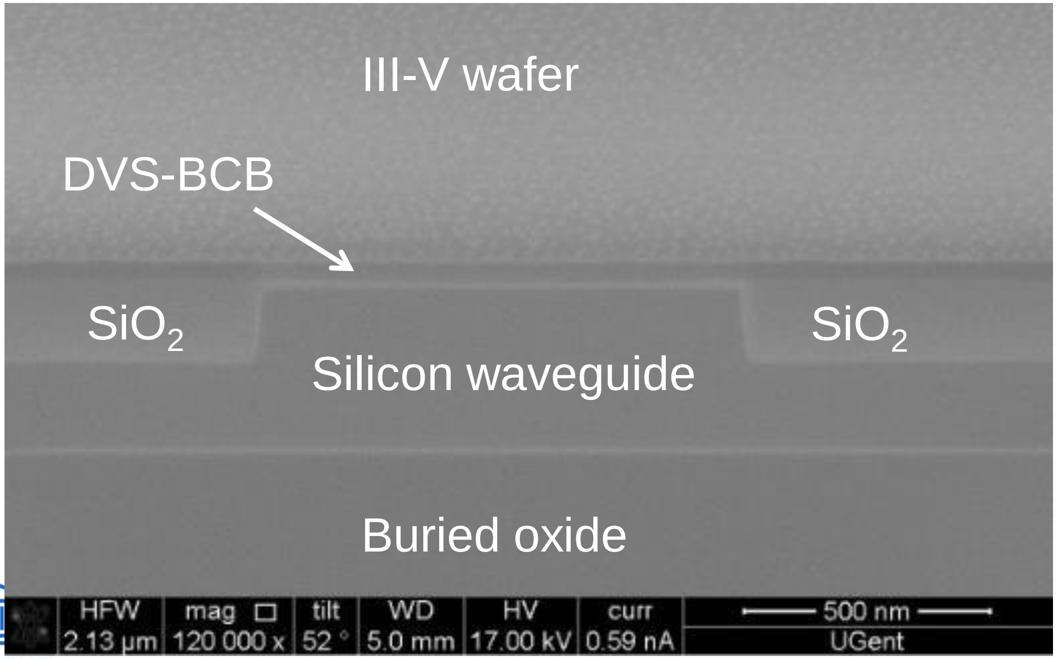
ALL PHOTONIC FUNCTIONS ARE THERE…

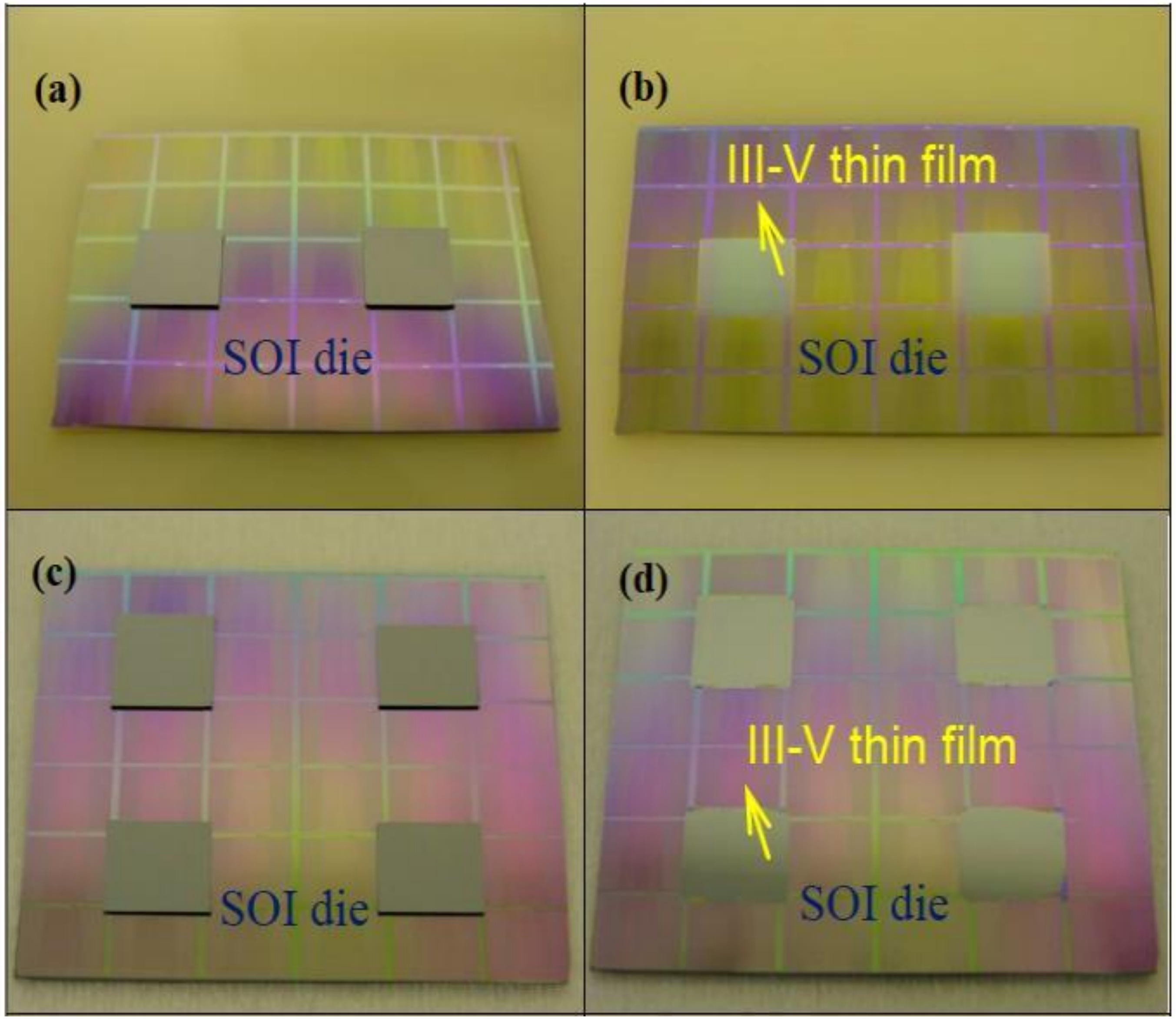


…except for the laser.

Laser integration on Silicon Photonics

Transfer III-V laser material to the silicon Photonic chip

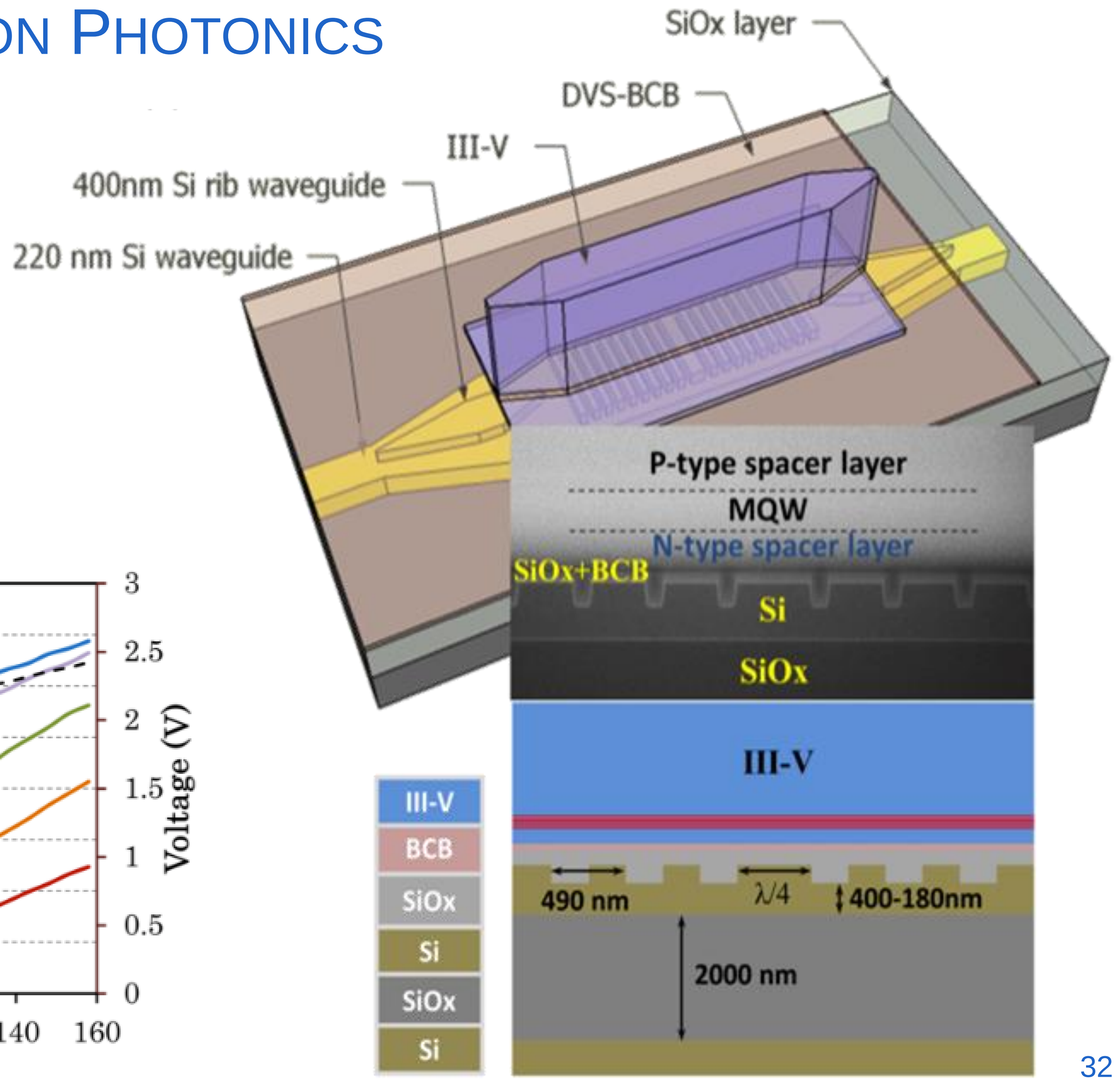


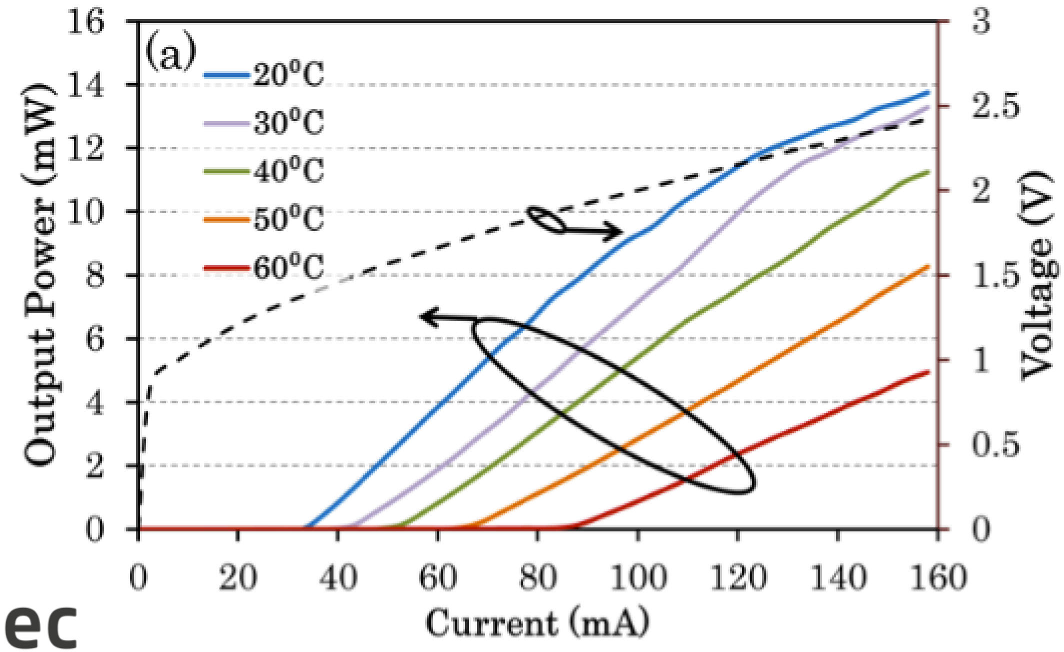


LASER INTEGRATION ON SILICON PHOTONICS

Different laser types

- DFB, DBR, microdisk,
- external cavity, multiwavelength, modelocked, …





SILICON PHOTONICS ENABLES LARGE SCALE PHOTONICS

>10000 optical functions on a chip

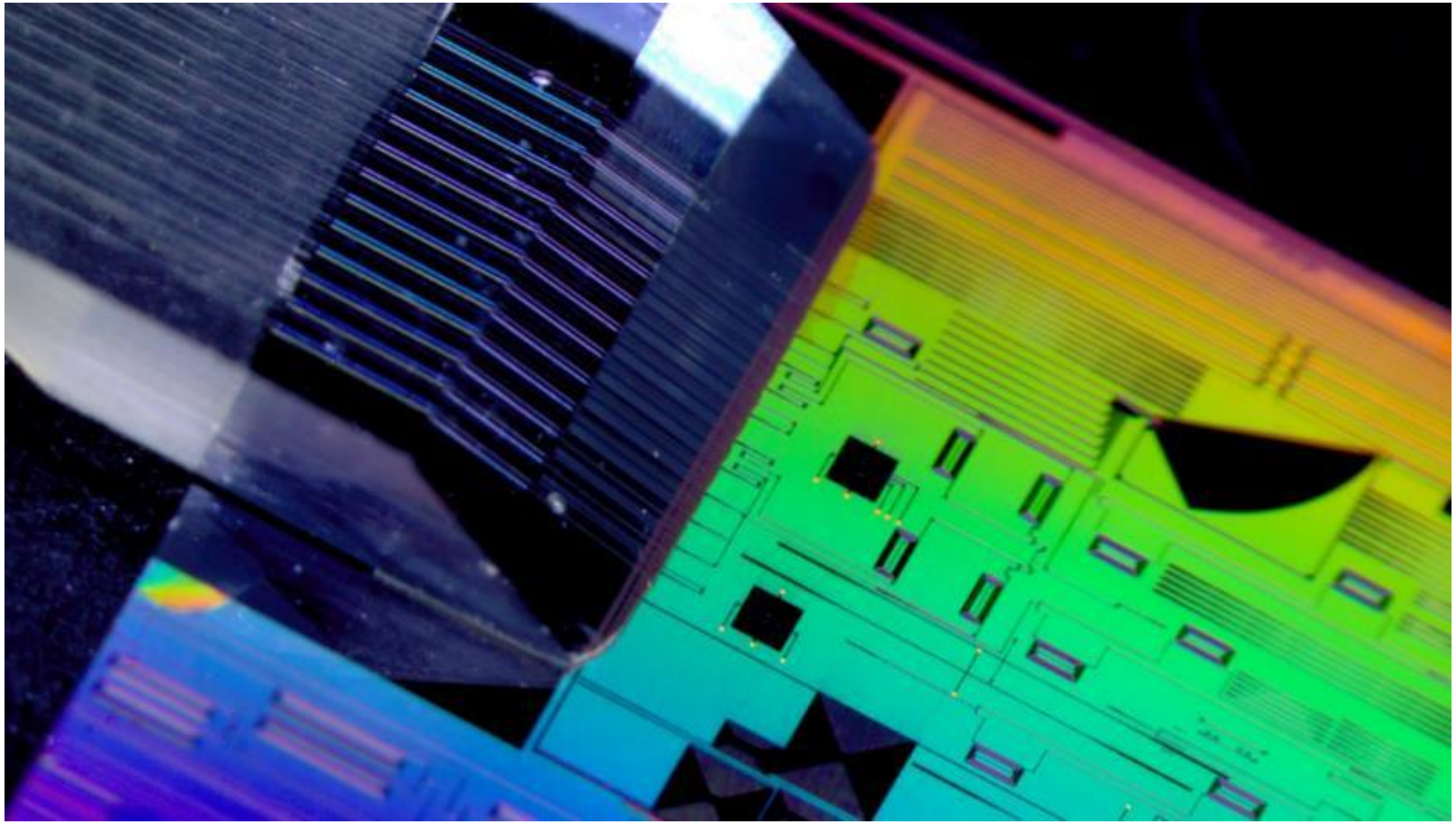
optical guiding, filtering,

detection and modulation

efficient fiber-chip coupling

external or integrated

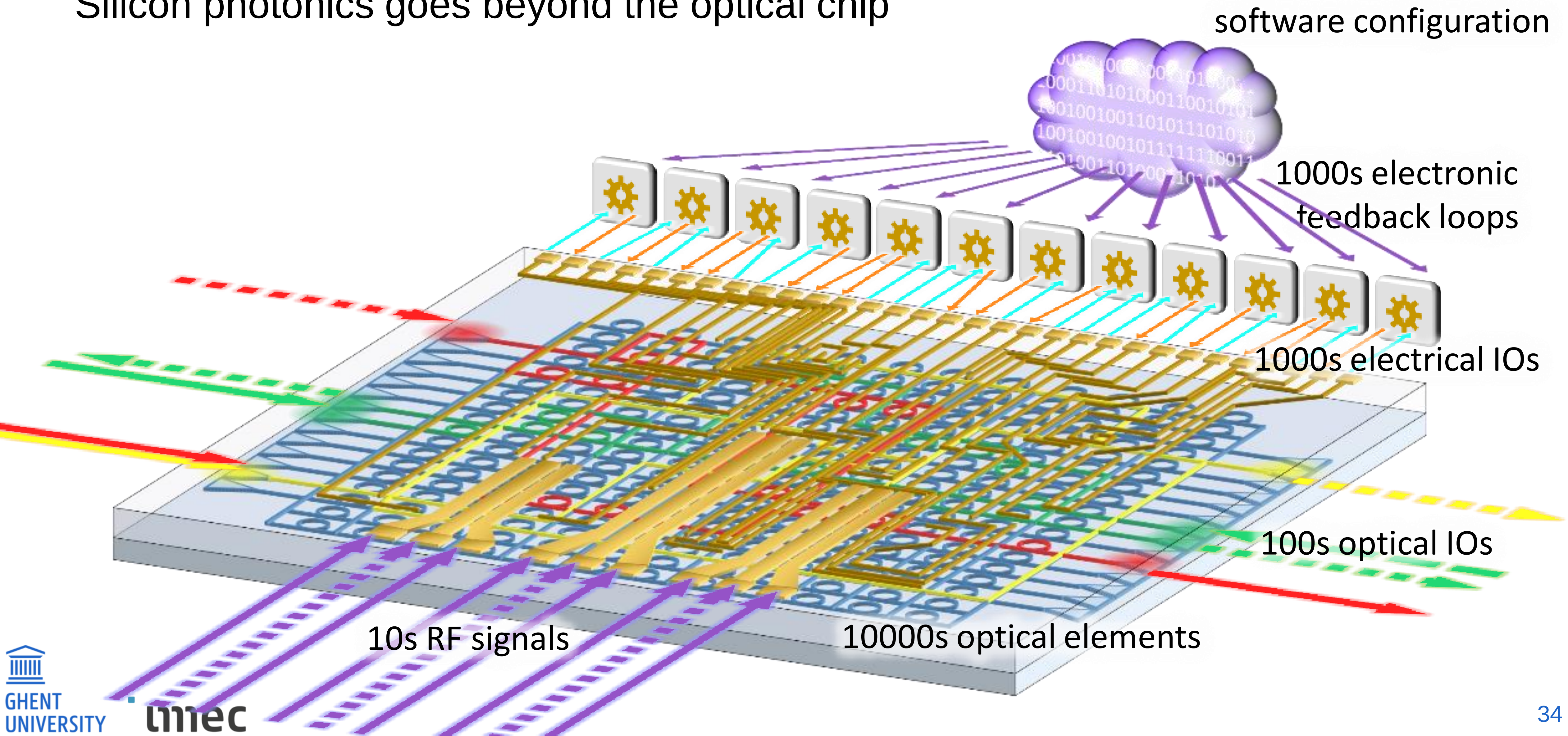
light sources



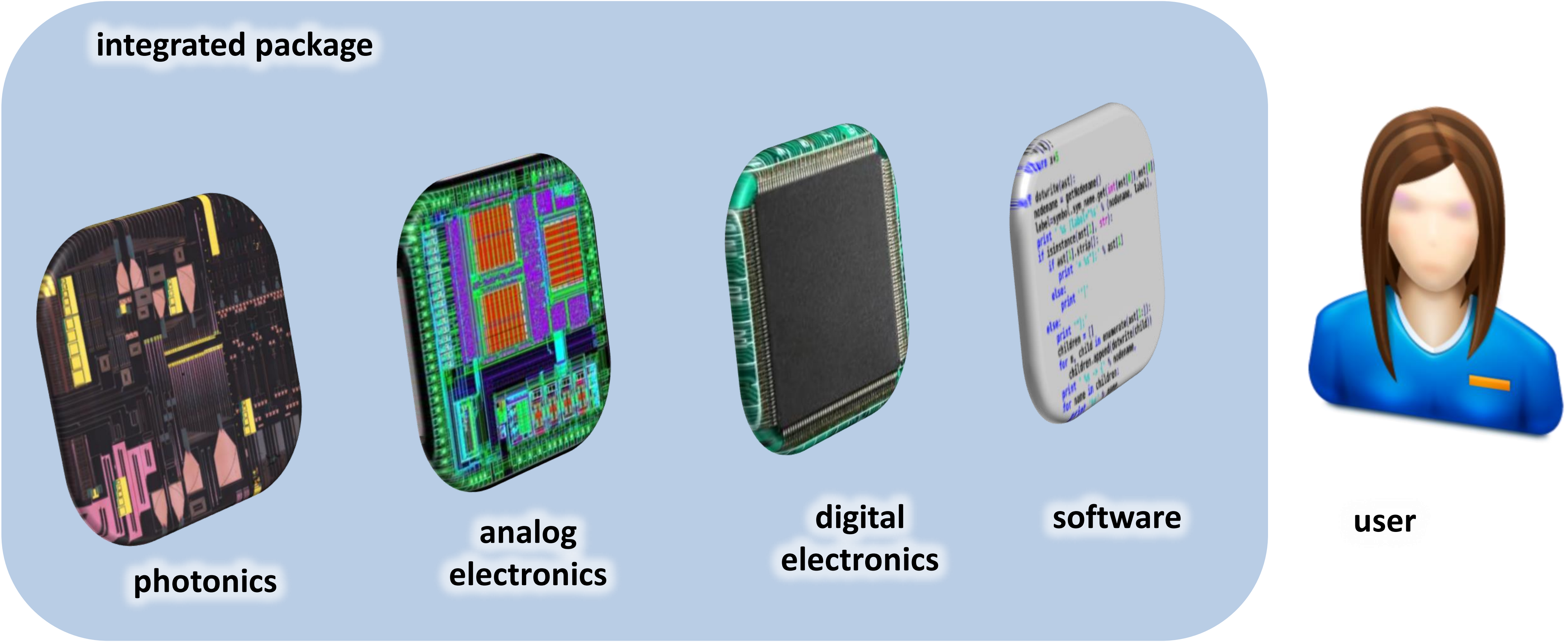
GHENT UNIVERSITY imec

MORE THAN JUST PHOTONS

Silicon photonics goes beyond the optical chip

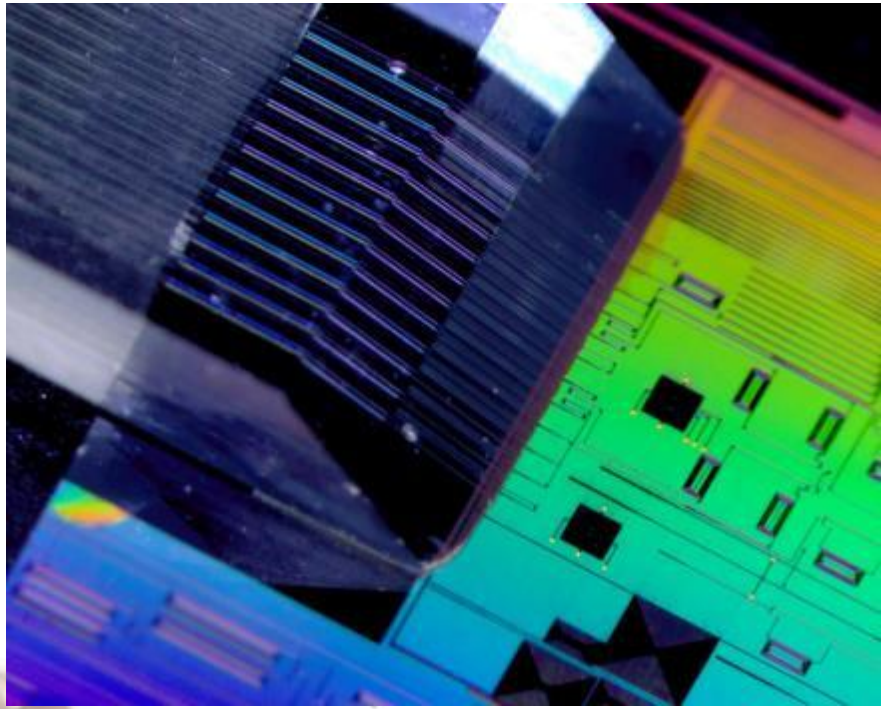


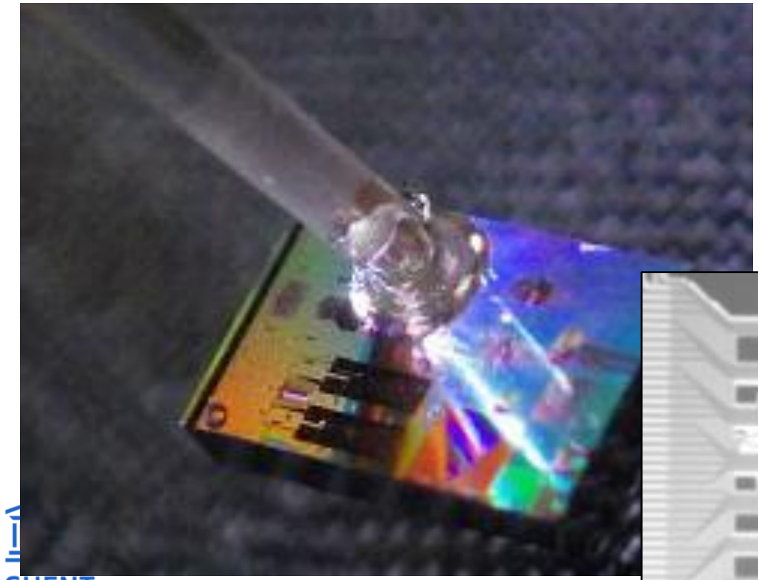
The Photonic Chip is Just a part of the system

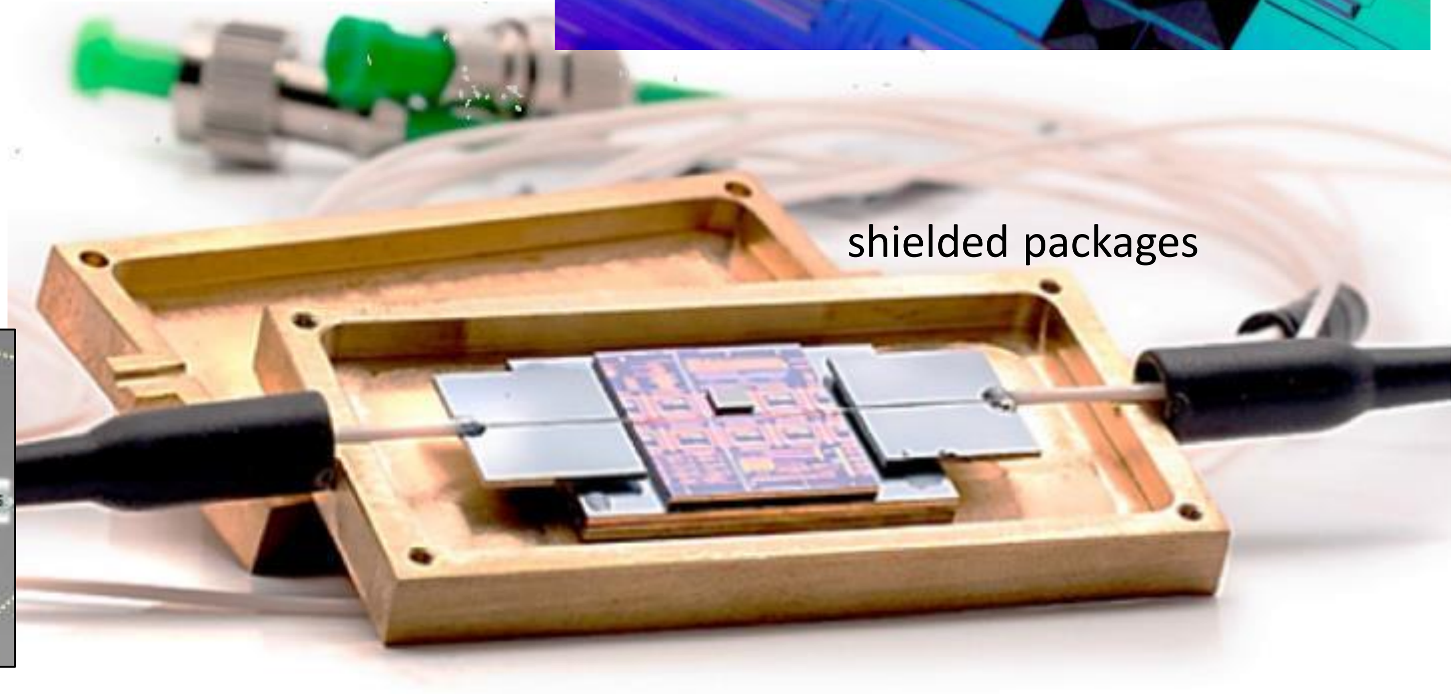


PACKAGING TECHNOLOGY

- Combining photonics and electronics
- Fiber interfaces
- RF connections
- Thermal and mechanical


fiber arrays

multi-core fibers



shielded packages

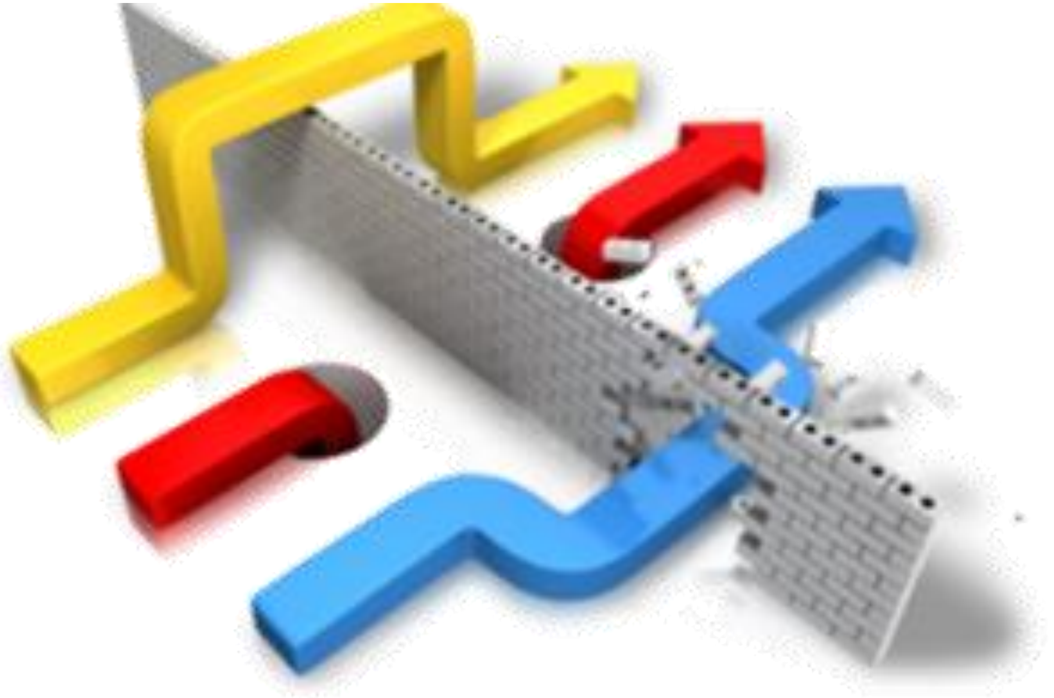
Fabless Silicon Photonics

Many fabless Silicon Photonics companies have emerged

- from direct collaboration with fabs (Luxtera, ...)
- starting from MPW (Caliopa, Genalyte, Acacia)

Established players are also partnering

- e.g. Finisar with ST
- Many keep their fab a secret



How to enter as a new (fabless) startup?

SMALL BUILDING BLOCKS → LARGE CIRCUITS

μm-scale building blocks

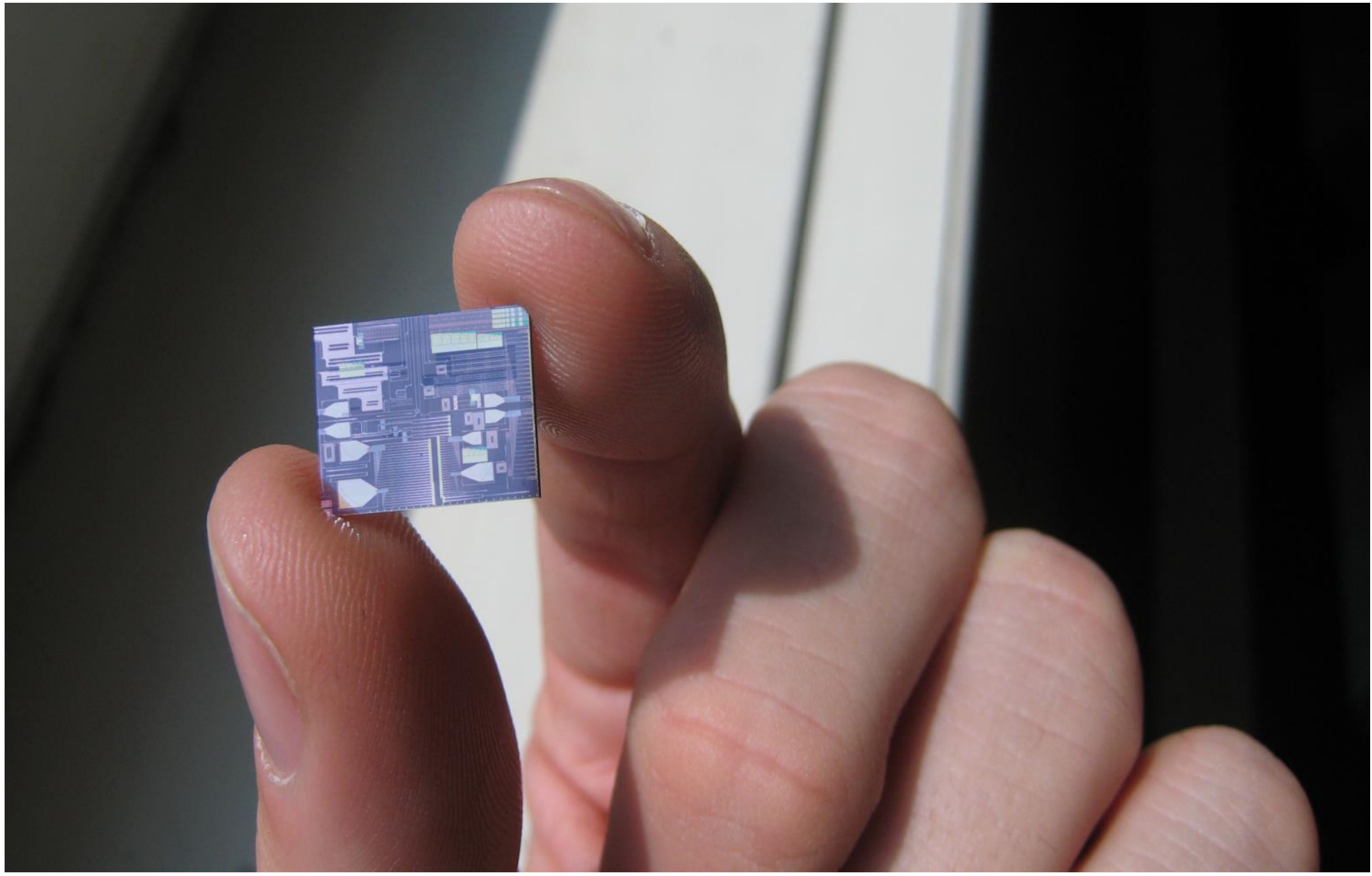
cm-scale chips

→ thousands – millions components

Photonics

Very Large Scale Integration

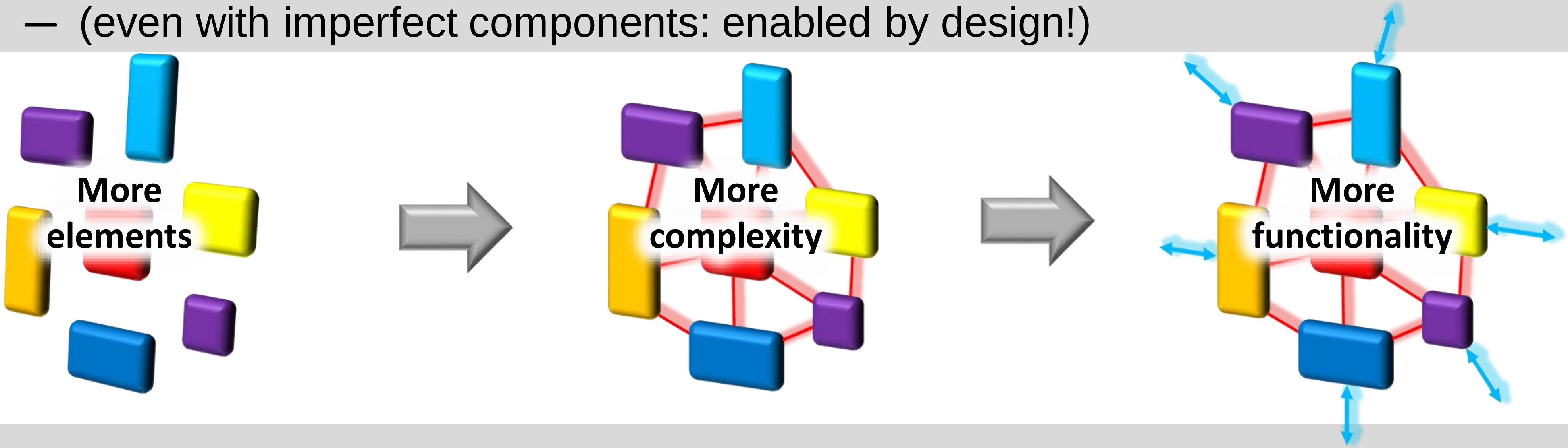
(VLSI)



COMPLEXITY AS AN ENABLER

Integrated Electronics

- billions of digital gates: unprecedented logic performance
- millions of analog transistors: unprecedented control
  - (even with imperfect components: enabled by design!)

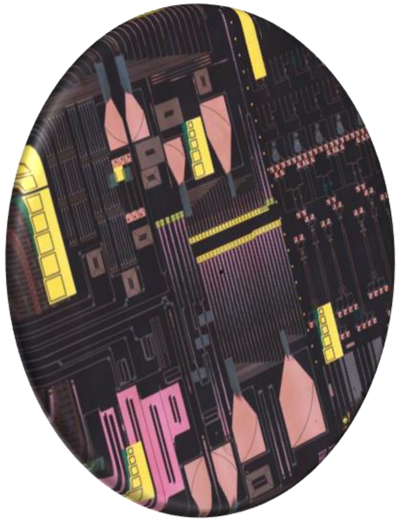


Integrated Photonics (Silicon Photonics)

- **technological potential** of 10000+ photonic elements on a chip
- not even scratched the surface of what this could do

PHOTONIC CIRCUIT DESIGN

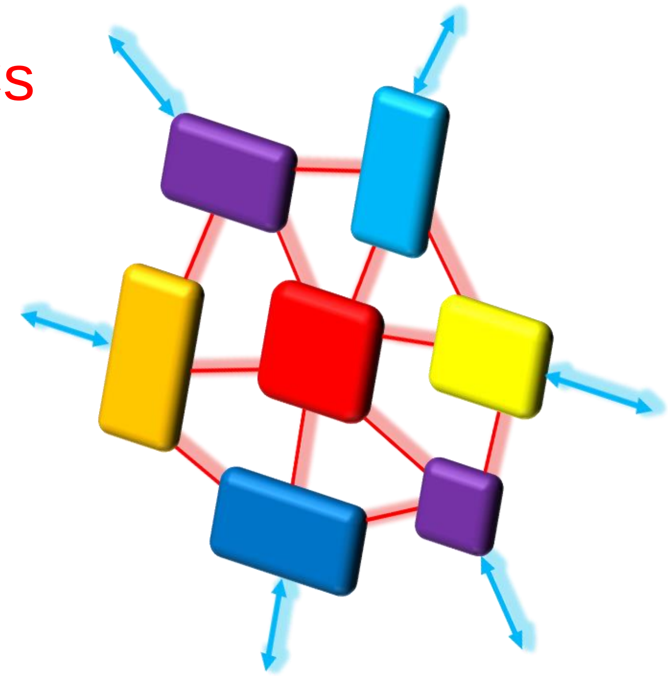
ENABLING COMPLEXITY IN PHOTONICS



Industrial PIC technology platforms (Si, InP, …)

- demonstrations of sensors, spectrometers, …
- commercial products

But: fairly simple circuits ~ 1970s ICs



More complexity is enabled by design methods

- Design capture: translating ideas to circuits
- Circuit simulation (electrical+photonic)
- Variability analysis on circuits
- Yield prediction and improvement

GHENT UNIVERSITY imec

Complex circuits ≠ complicated building blocks

You can do a lot with a few building blocks

Electronics: Transistors, Resistors, Diodes, …

Photonics: Waveguides, Directional couplers, …

Complexity emerges from connectivity

But you need to support complexity

- Accurate models
- Variability
- Parasitics

GHENT UNIVERSITY imec

DESIGNING PHOTONIC INTEGRATED CIRCUITS

Can we learn from electronic ICs?

- Millions of analog transistors
- Billions of digital transistors
- Power, timing and yield
- **First time right designs**

- Very mature Electronic Design Automation (EDA) tools!
- A well established design flow

Can we repurpose this for photonics?

Design Environments are Emerging

Combinations of Photonics Design and EDA

Physical simulation combined with circuit design

Physical and functional verification

First PDKs with basic models

LUCEDA PHOTONICS

Mentor Graphics®

CST

VPIphotonics DESIGN AUTOMATION

SYNOPSYS®

Photon Design®

PhoeniX Software
Solutions for Micro and Nano Technologies

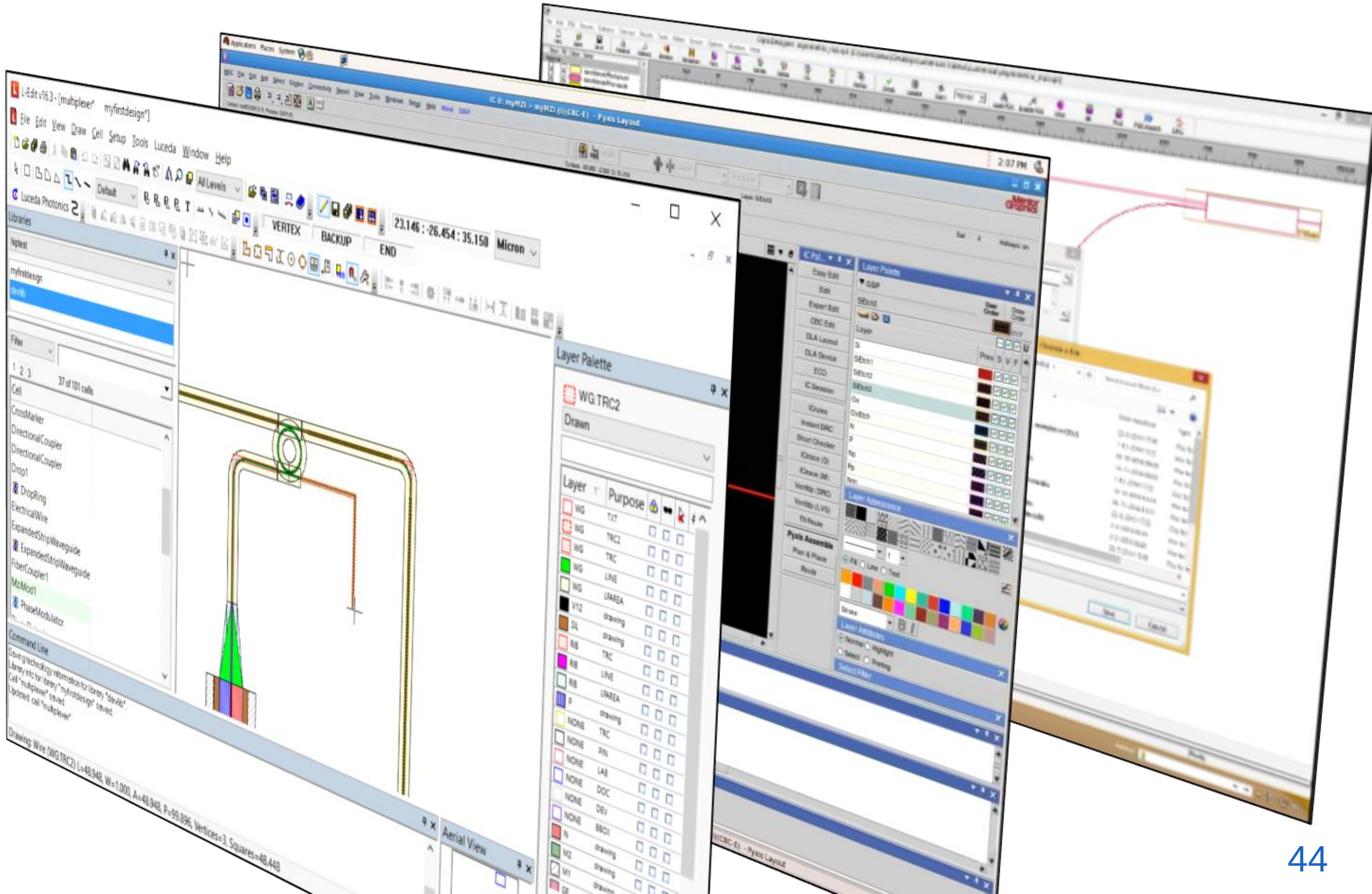
lumerical

cādence

GHENT UNIVERSITY

imec

Optiwave



WHAT IS A DESIGN FLOW?

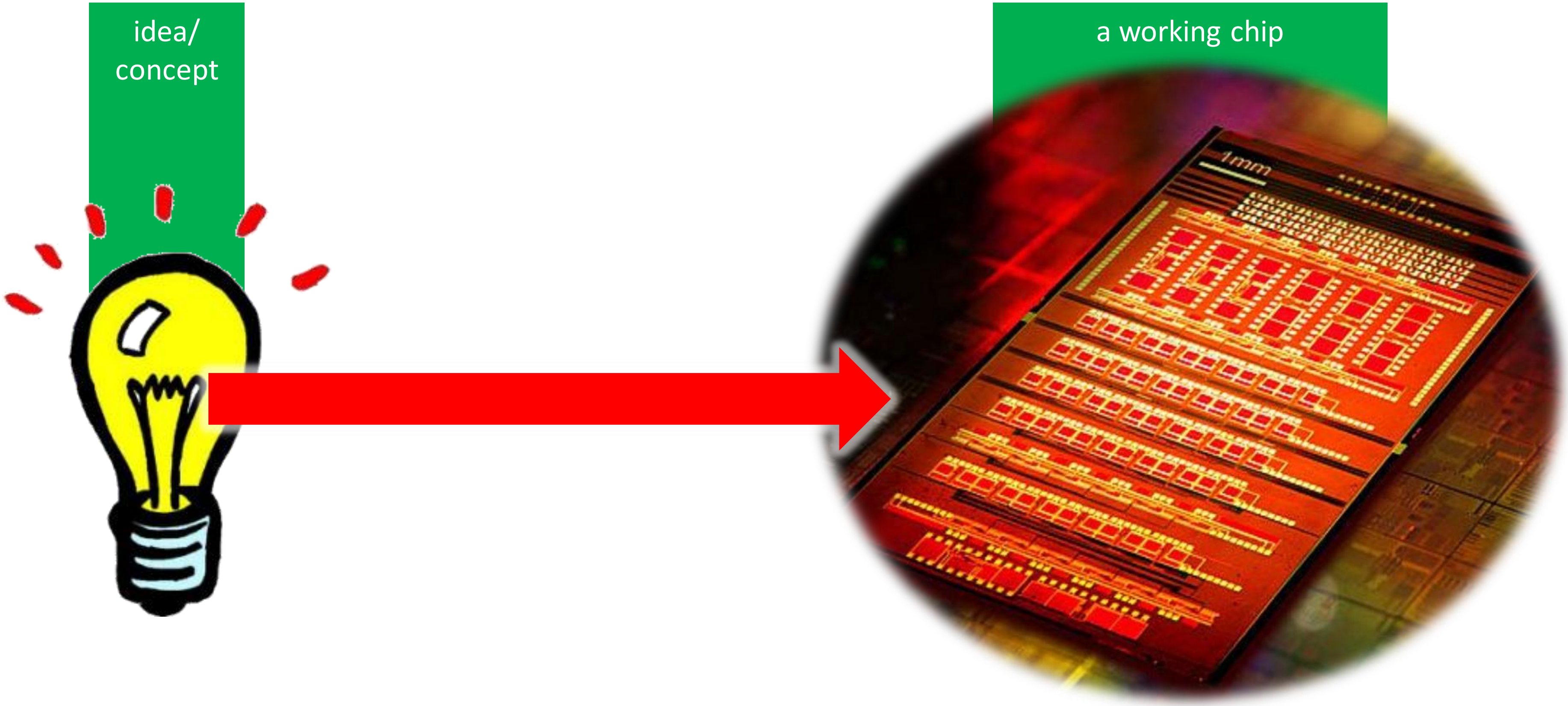
"Design is the creation of a plan or convention for the construction of an object or a system"

Design Flow

"a repeatable pattern of activity, usually involving multiple tasks with a specific set of outcomes"

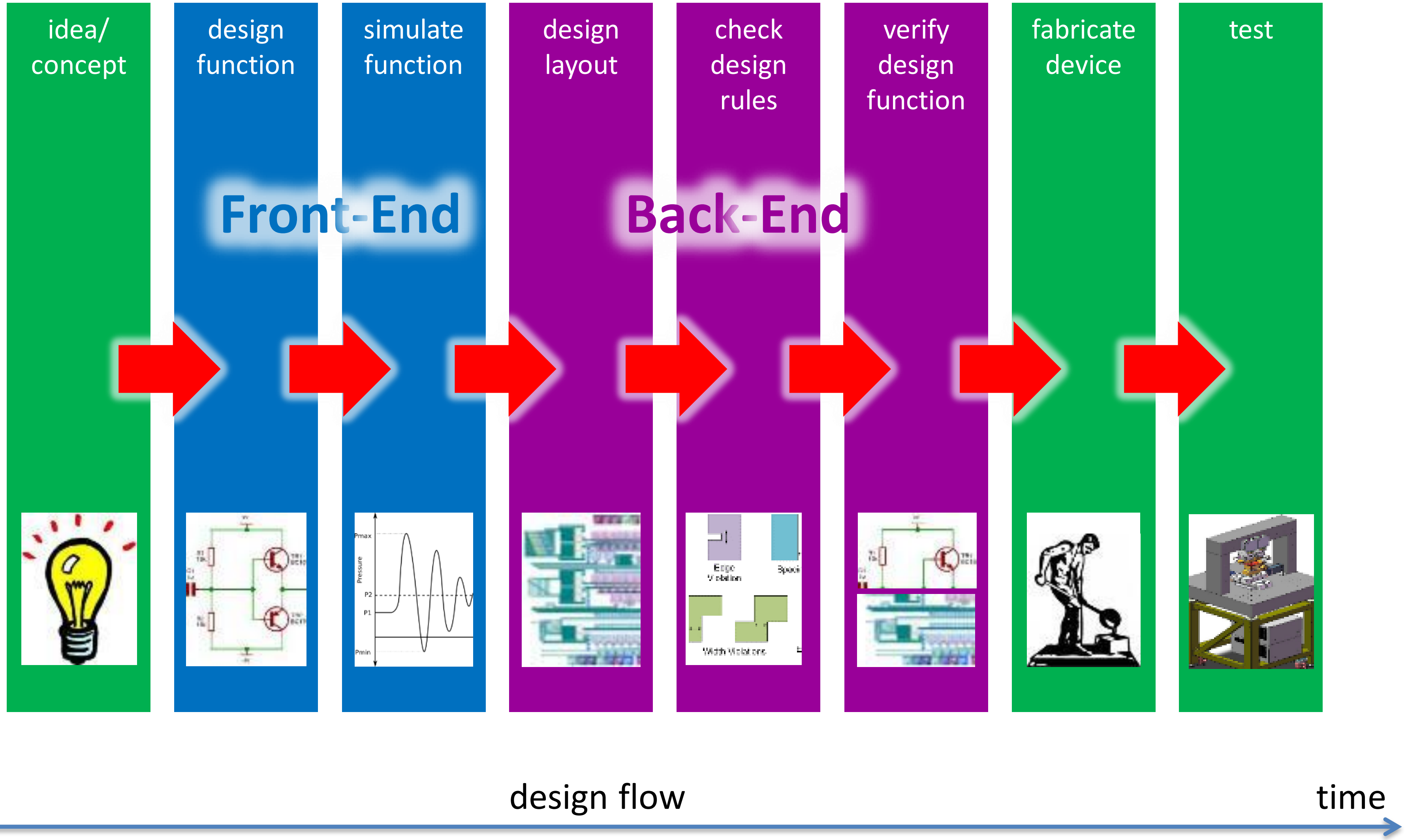
GHENT UNIVERSITY imec

WHAT IS THE PURPOSE OF A DESIGN FLOW?



to translate an idea into a **WORKING** chip.

A TYPICAL DESIGN CYCLE



A GREAT IDEA?

idea/
concept

Questions to be asked

- What should my device do?
- What are its operational principles?
- How well should it perform?
- Where/How will it be used?

To go from an idea to a design, you need

SPECIFICATIONS

design flow

time

GHENT UNIVERSITY

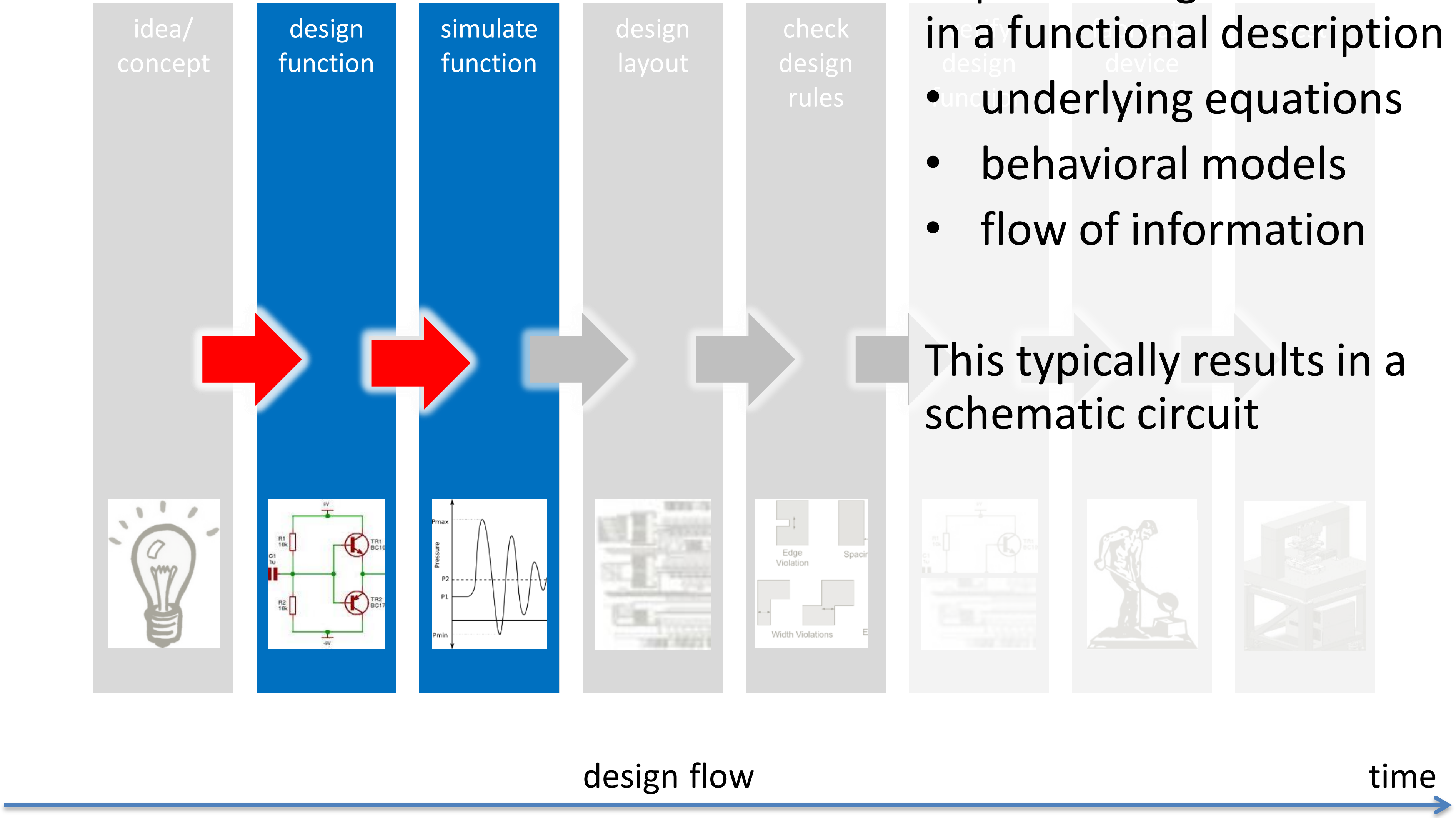
imec

DESIGN CAPTURE AND SIMULATION

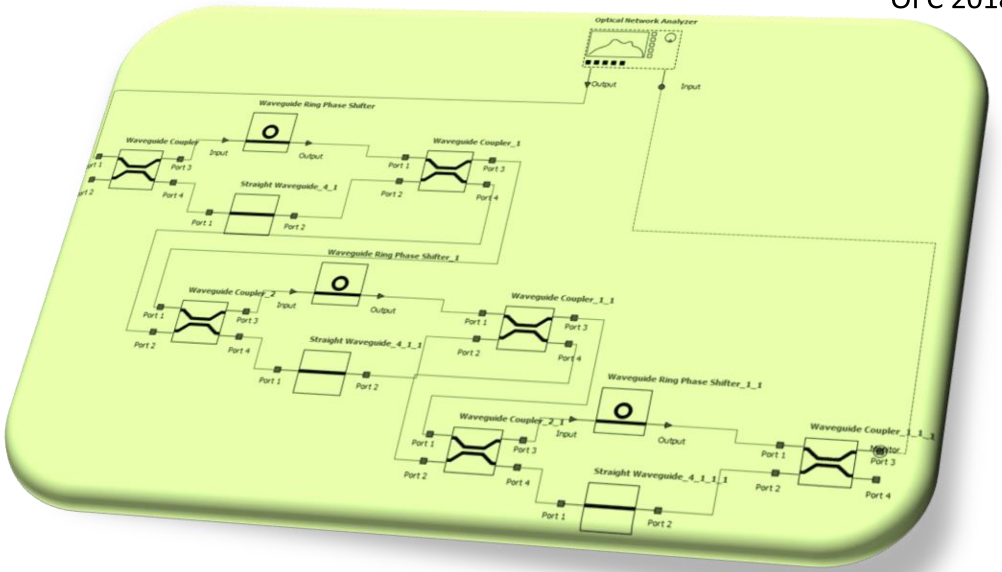
Capture design intent in a functional description

- underlying equations
- behavioral models
- flow of information

This typically results in a schematic circuit



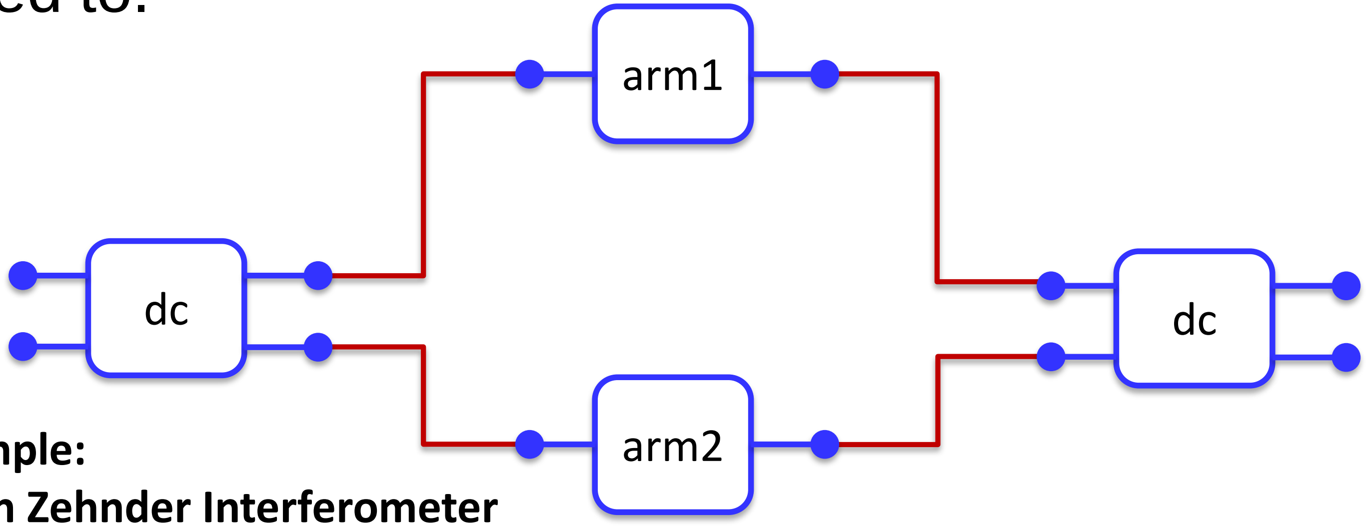
DESIGN CAPTURE



Select/construct functional blocks

Connect them together

- **Netlist**:
  list of connections (“Nets”) and which components the nets are attached to.
- **Schematic**:
  graphical representation of a netlist, with placements



Example:
Mach Zehnder Interferometer

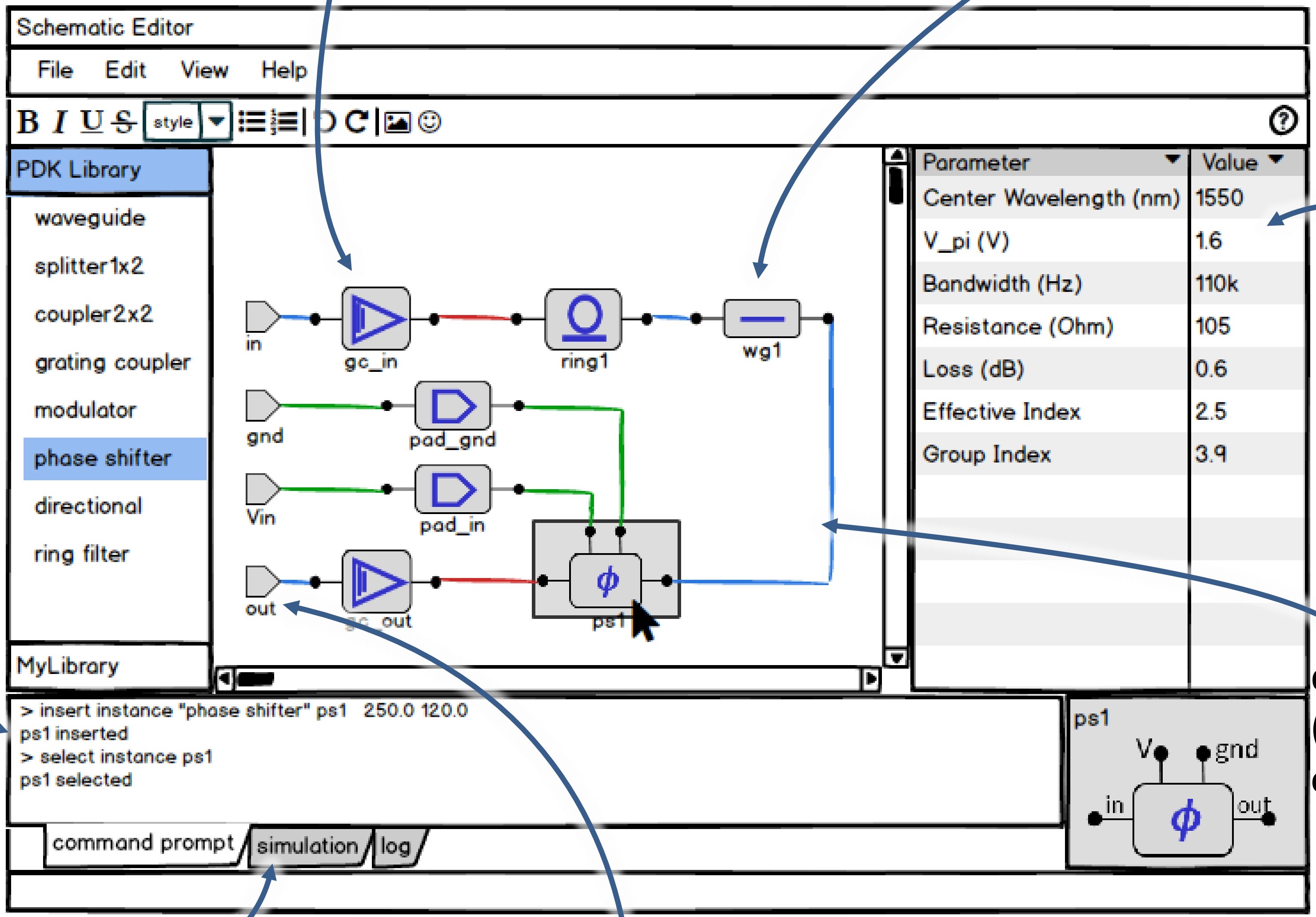
SCHEMATIC EDITOR

drag and dropping components and drawing connections

make waveguides explicit if needed

component libraries

parametrization



different connections (waveguides, direct optical, electrical)

scriptability

interface to circuit simulation

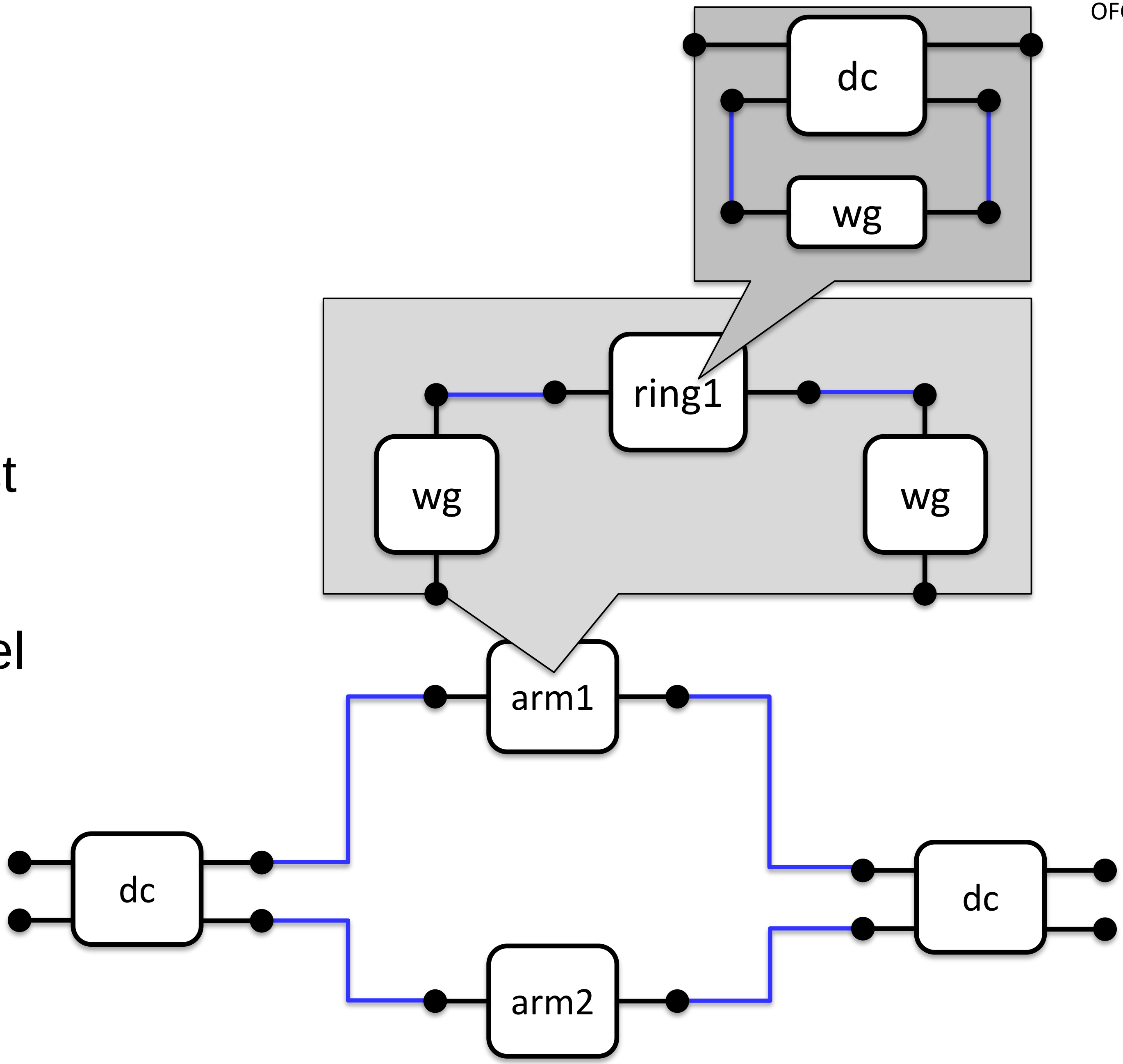
specify I/O ports

GHENT UNIVERSITY imec

HIERARCHY

Netlists are hierarchical

- Hierarchical cells:
  contain another netlist
- Atomic cells:
  contain a circuit model



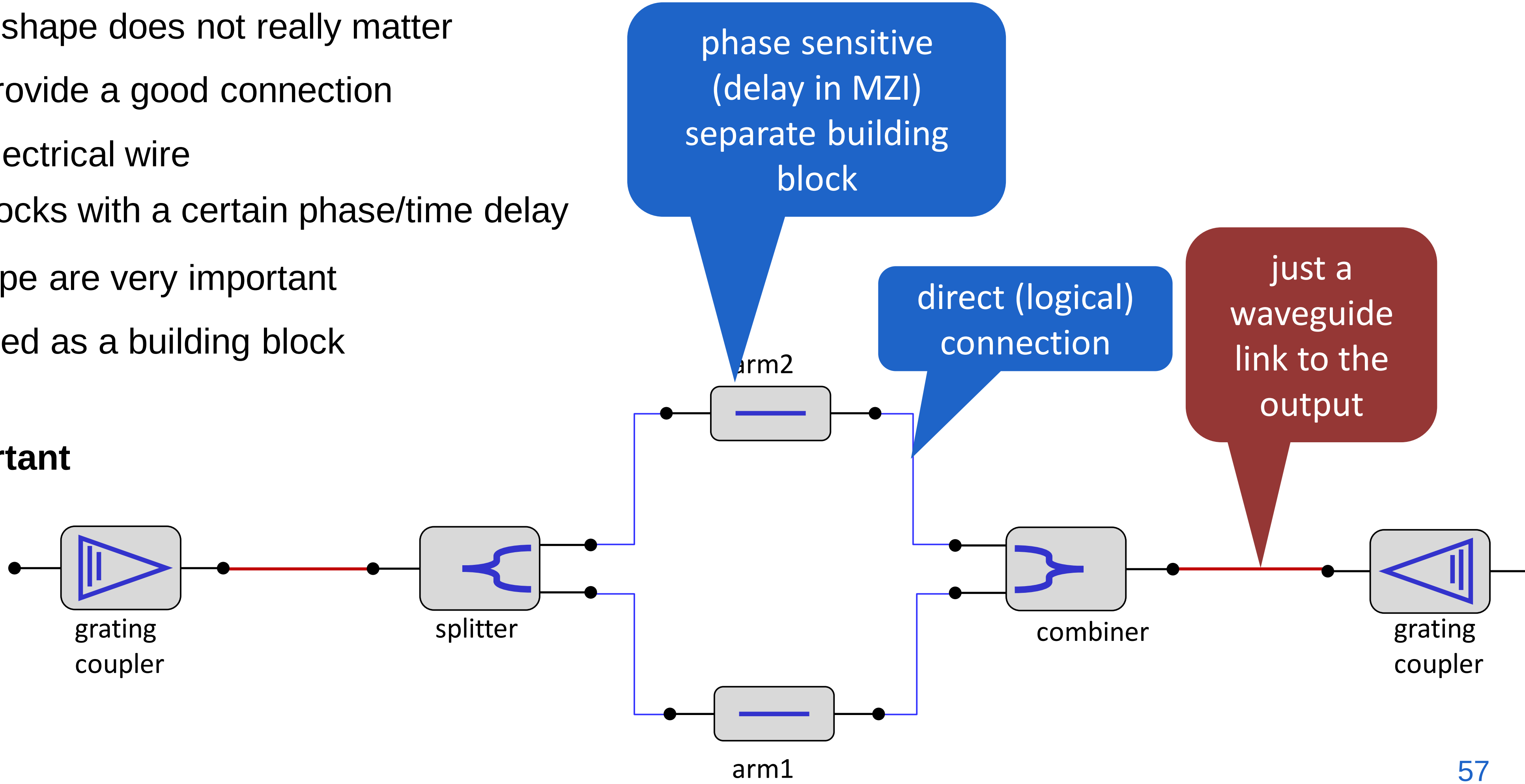
Example:
Ring-Loaded Mach Zehnder Interferometer

Waveguides in Photonic Schematics

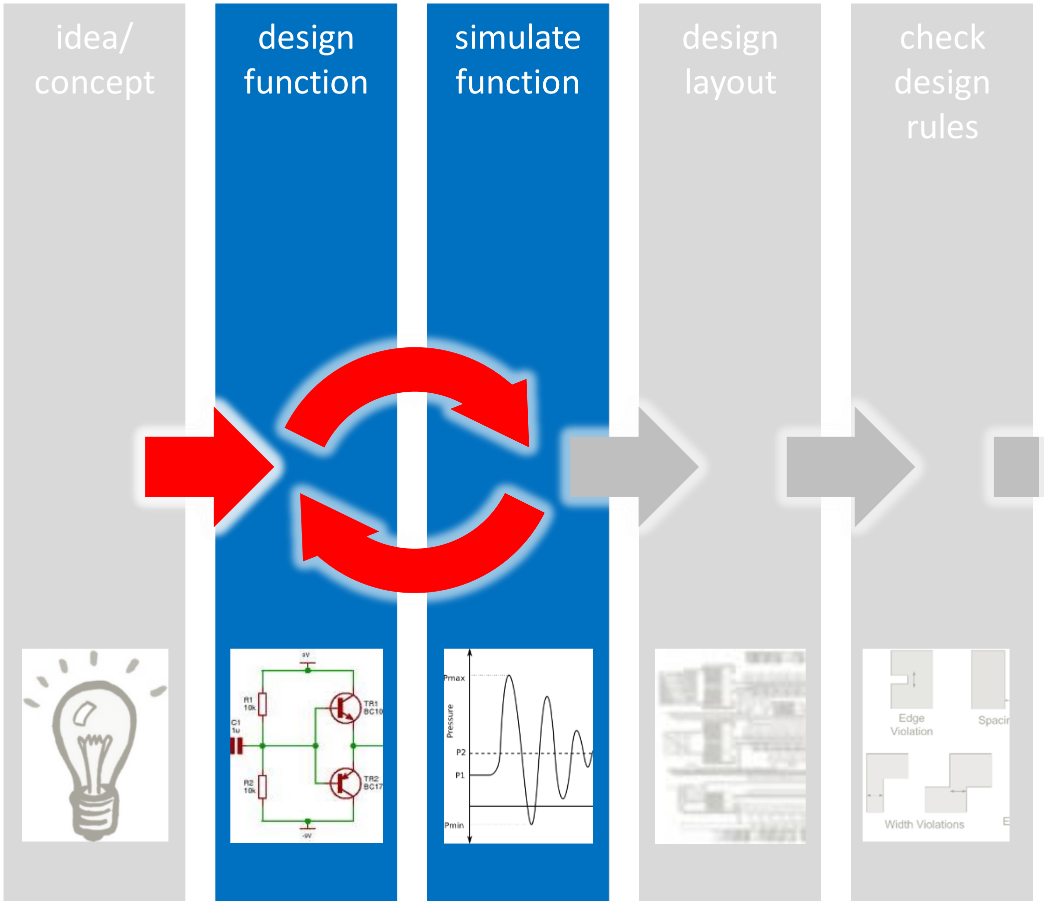
What are waveguides?

- Simple connections between building blocks
  - the length and shape does not really matter
  - it should just provide a good connection
  - similar as an electrical wire
- Functional building blocks with a certain phase/time delay
  - length and shape are very important
  - should be treated as a building block

The distinction is important



DESIGN CAPTURE AND SIMULATION



Capture design intent in a functional description

- underlying equations
- behavioral models
- flow of information

This typically results in a schematic circuit

Simulated at an abstract level

Optimization: an iterative process

design flow

time

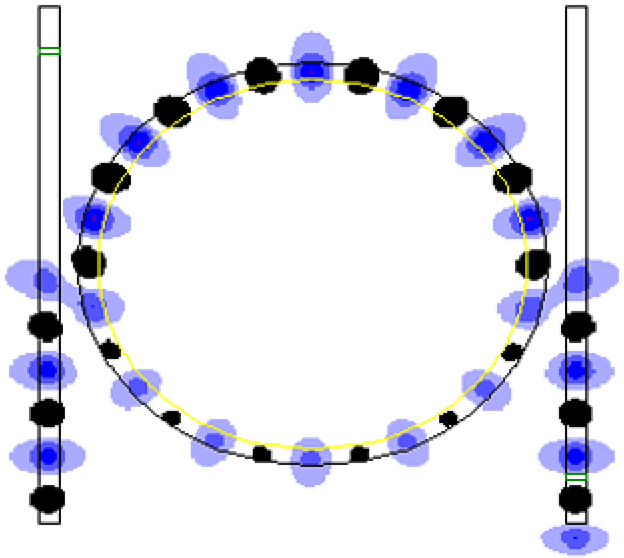
COMPONENT SIMULATION ≠ CIRCUIT SIMULATION

Physical models

Accurate, slow

Based on actual geometries

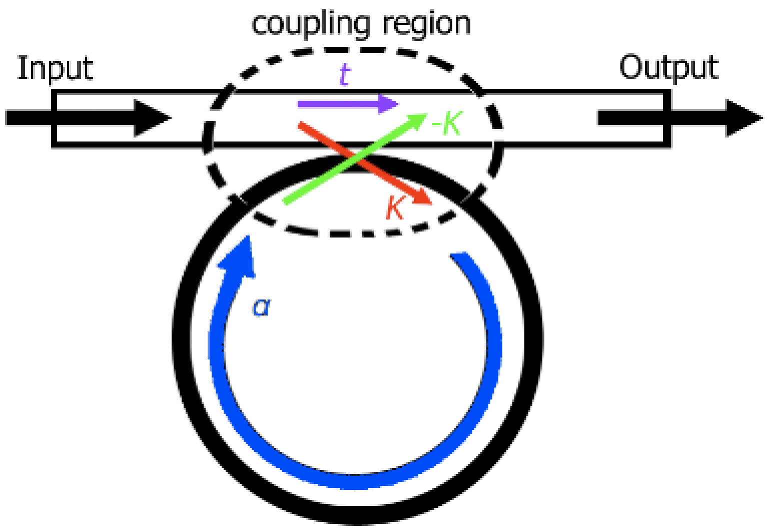
Best model = reality



Circuit simulations

Approximate, fast

Based on functional description



behavioural models

MODELS

Should allow simulation in a larger circuit

- based on equations
- based on measurement data
- based on EM simulations

Photonics: Nothing really standardized

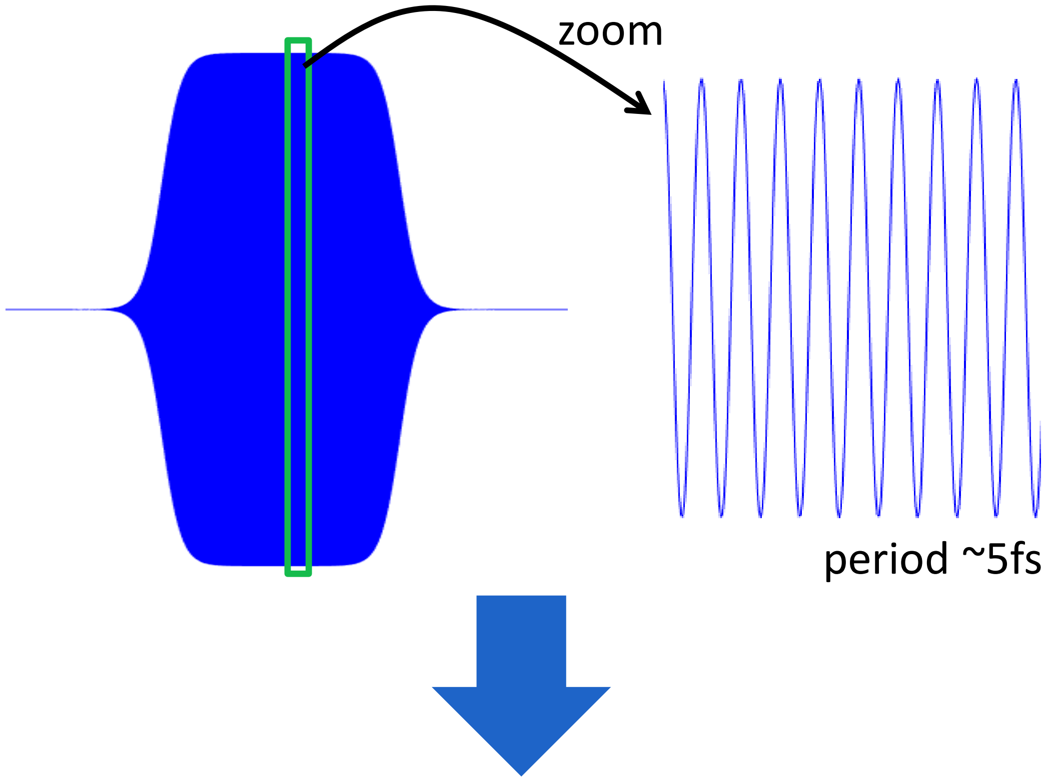
- No standardized simulation method
- No standard model description
- No standard signals

Optical Vs. Electrical Circuit Simulation

optics = electric… at very high frequency

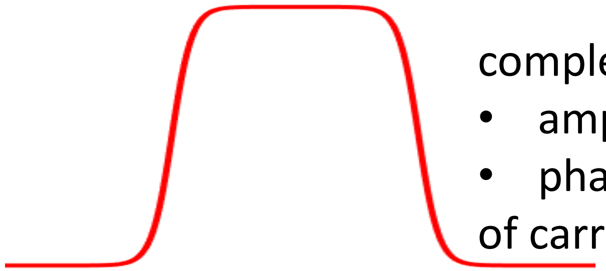
- ultra-small time steps (fs)
- ultra-long simulations (10^{12} time steps)
- high-bandwidth signals (200THz)

impractical.



Solution: analytic signal

= complex amplitude on carrier



complex envelope
- amplitude
- phase

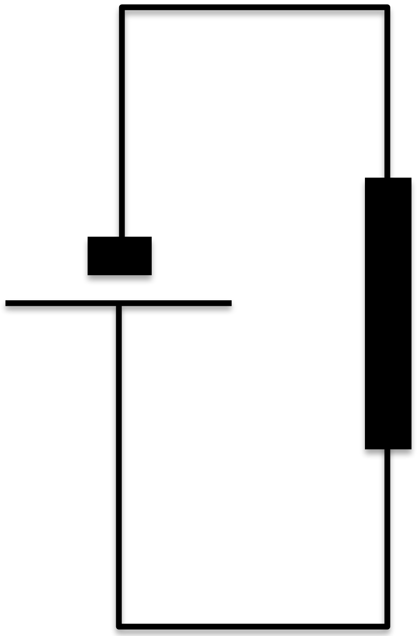
of carrier

Photonic Circuit Simulations

Same as electronics?
No. Photonics does not fit in Spice

Effort-flow systems

Electrical	Voltage	Current
Fluidic	Pressure	Flow
Thermal	Temperature	Heat Flow*
Mechanical	Force	Motion
Photonic?	E-field	H-field



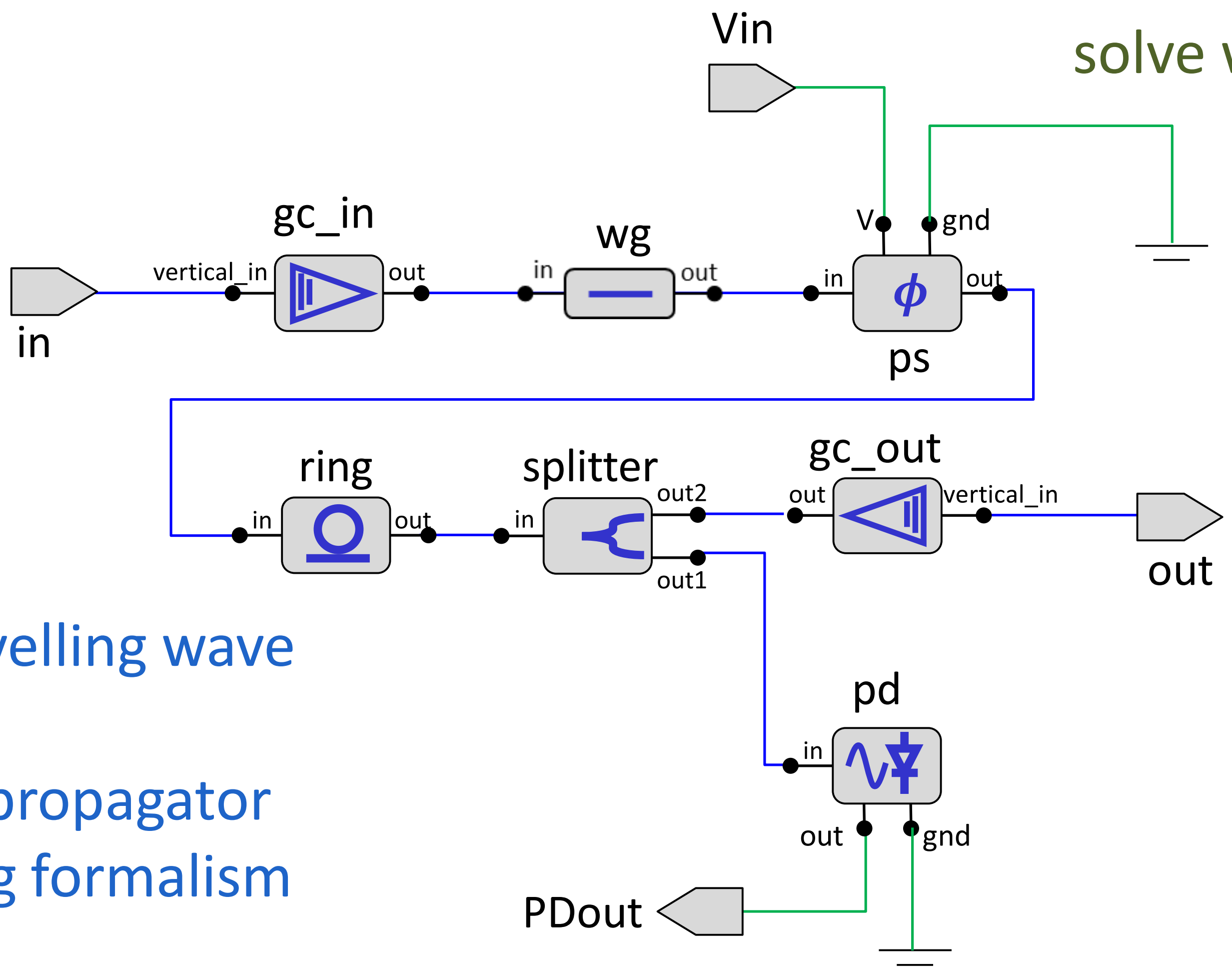
Not the best formalism for photonics (too high frequency, much more than an RF wave)

Wave scattering formalism ≠ Effort Flow Formalism

electrical and optical is not the same

electrical:
- nets with a voltage potential
- current flowing through terms

solve with SPICE



optical:
- ports
- links with a travelling wave
- bidirectional

solve with signal propagator or wave scattering formalism

GHENT UNIVERSITY imec

Linear Photonics: Scatter matrices

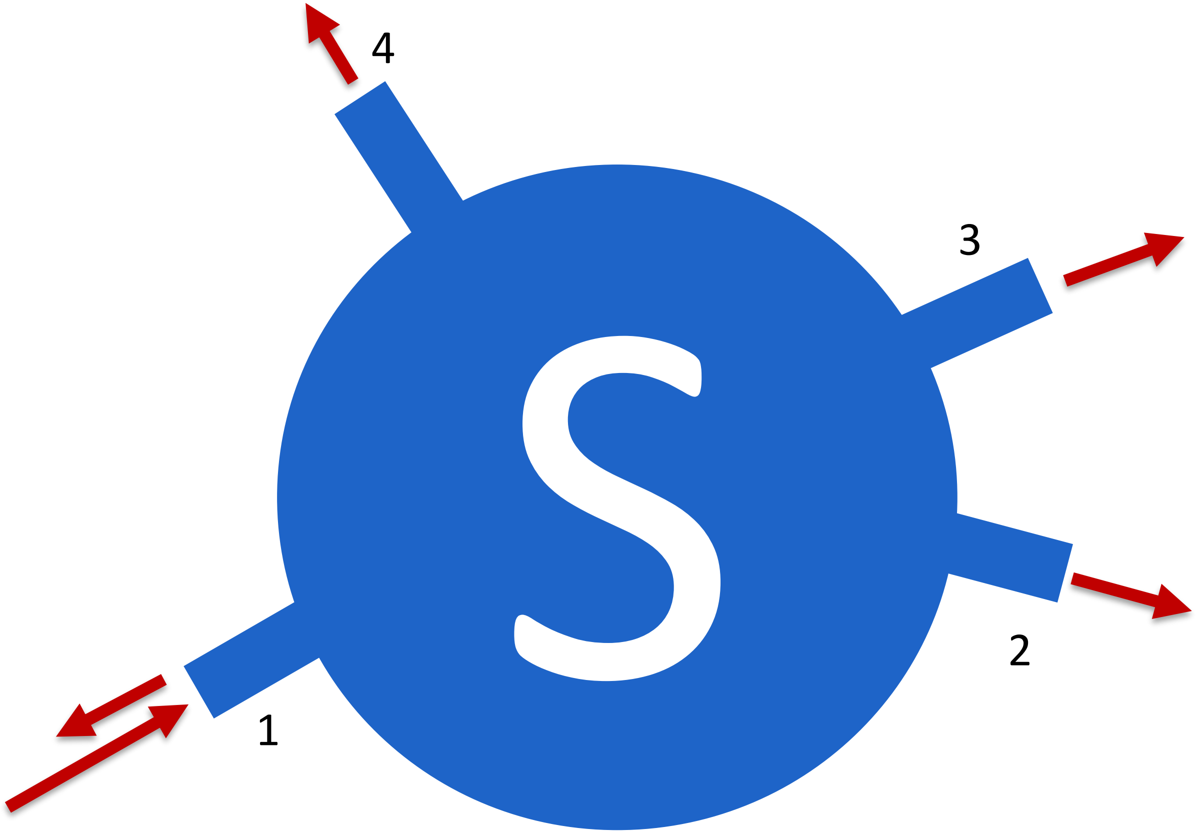
Generalized reflections of a propagating wave

Linear coupling between all 'ports'

- waveguides
- modes

$$S_{21}(\omega)$$

to   from   frequency (wavelength) dependent



Includes reflection!

WHAT IS A PORT OF A WAVEGUIDE COMPONENT?

Orthogonal states

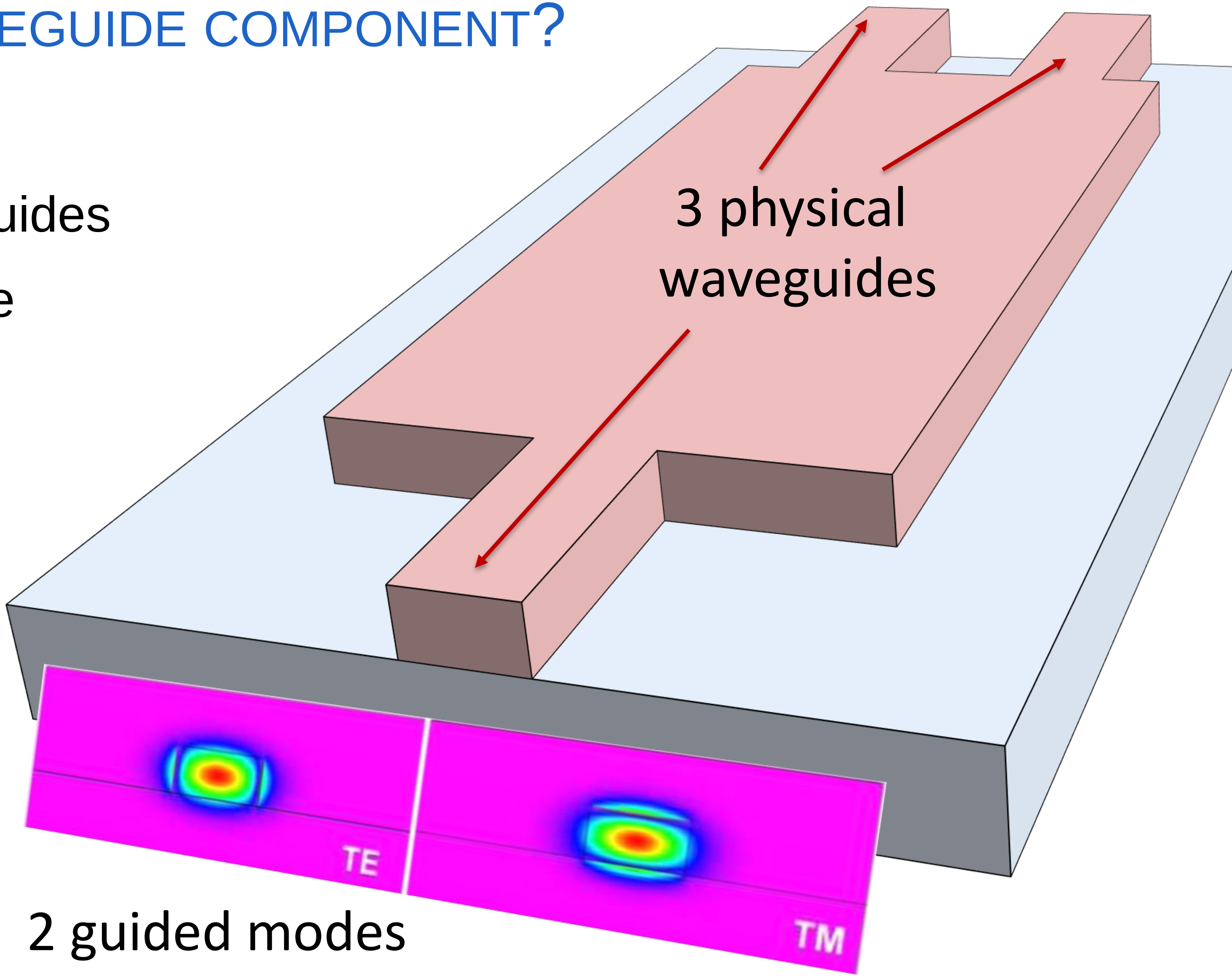
- Physically separated waveguides
- Each mode in the waveguide

Example: 6 “ports”

6x6 S-matrix

In practice:

Only use the relevant modes (rest is “loss”)



FREQUENCY DOMAIN OPTICAL CIRCUIT SIMULATOR

Frequency domain

- Linear systems
- Described by scatter matrices (S-parameters)

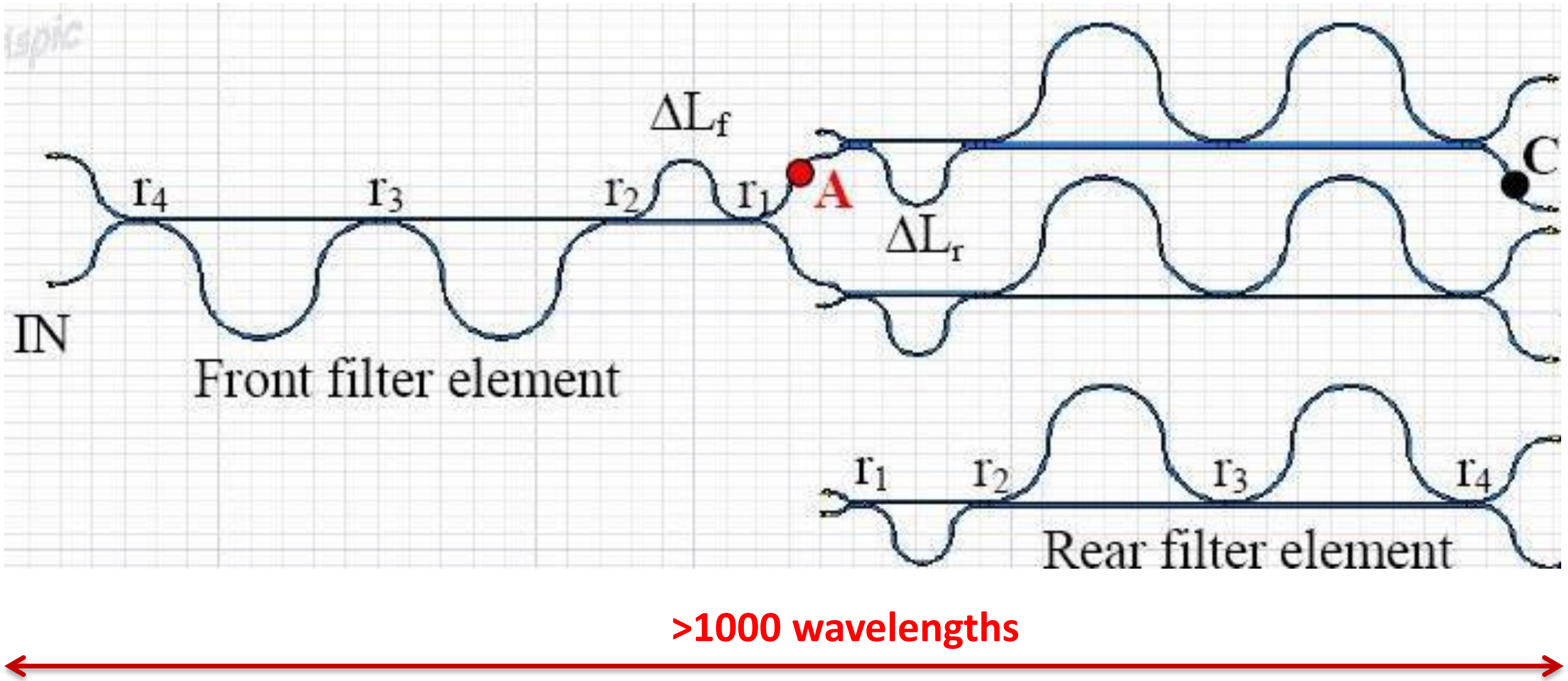
Circuit is solved as a single matrix (similar as RF)

Pro:

- Very fast
- Large circuits

Con:

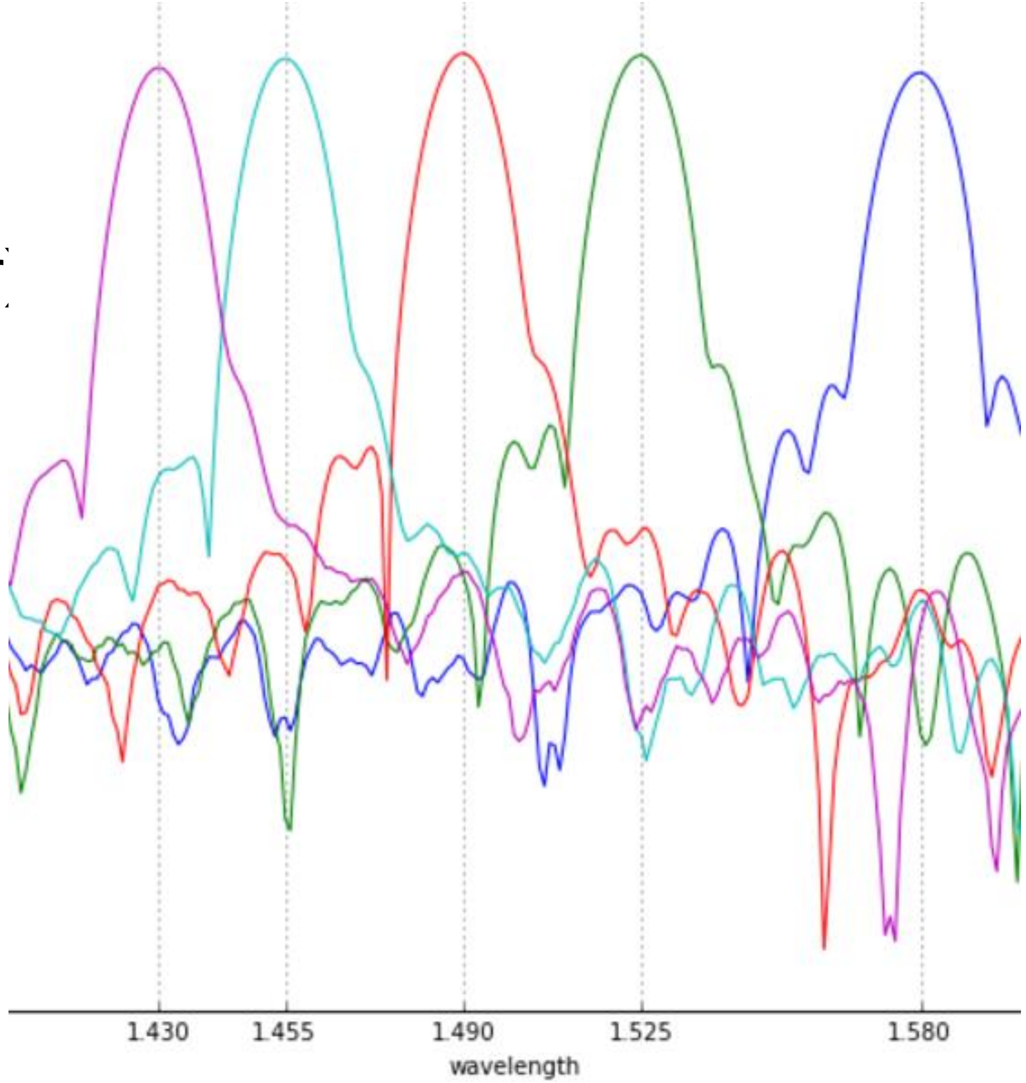
- No nonlinear effects



FREQUENCY DOMAIN SIMULATIONS

Frequency domain simulations are very useful for calculating

- Insertion losses
- Backreflections
- Dispersion (wavelength dependent behavior)
- Wavelength filter response



and can also be extended to model

- Slowly varying effects
- Certain optical nonlinearities

TIME DOMAIN OPTICAL CIRCUIT SIMULATION

Calculate time response of a circuit

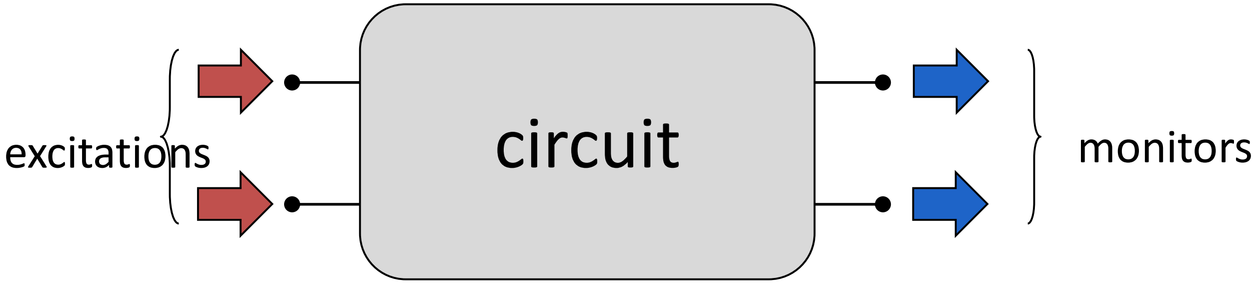
- to a stimulus (or combination of excitations)
- at certain output monitors
- using discrete time steps

Pro:

- Fast
- Large circuits

Con:

- Slower than frequency domain
- Only response to specific stimulus



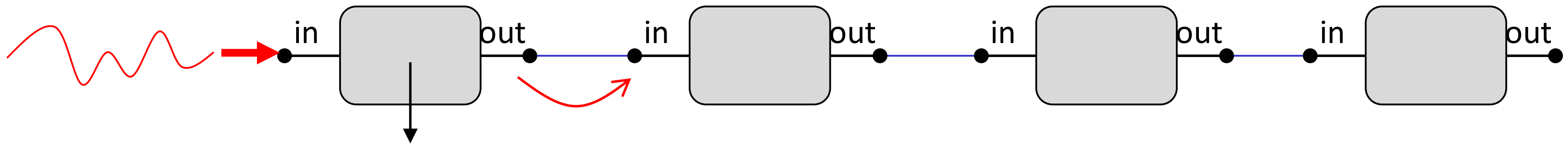
Time-Domain Optical Circuit Simulation

Nodes

- connected by signal lines (bidirectional)
- an internal state
- an algorithm to calculate output from inputs and internal state (differential equations, coupled-mode theory, custom code)

every time step, in each node:

- Input signals of last time step are read
- Internal state is updated
- Output signals are generated

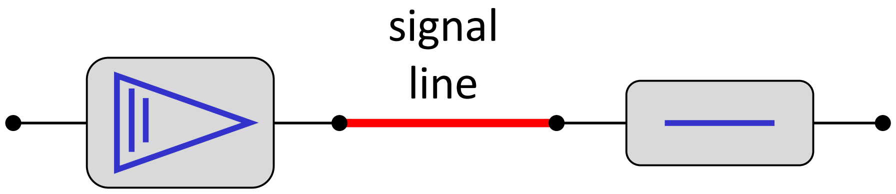


$$out[t] = f(in[t-1])$$

N steps delay

Optical Signals

An optical link carries an an optical signal…

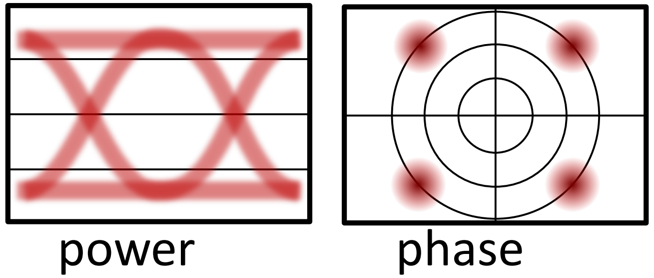


$$2 \times 2 \times N \times M$$

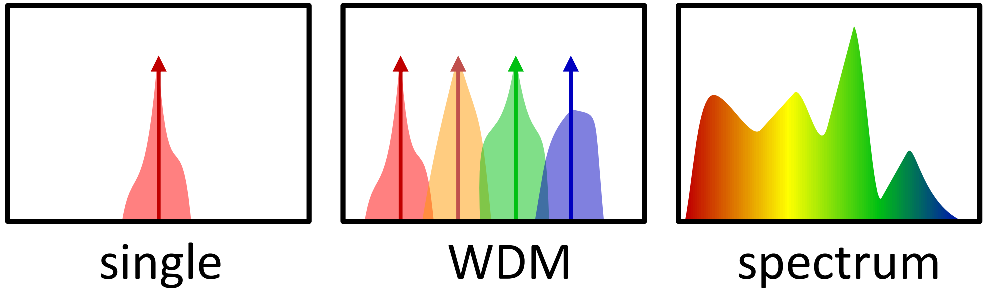
not all simulators
support all combinations

two directions

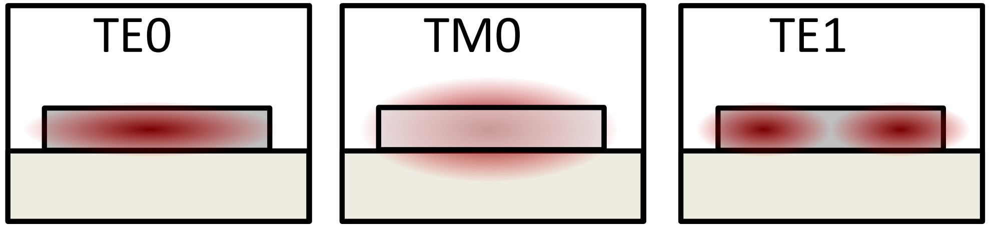
complex number



wavelength:
N channels



mode/polarization:
M modes



Optical Signals: Example

Example: Single-λ link

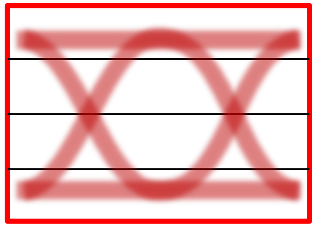
- One direction
- One wavelength
- On-off-keying: power
- One mode: TE

$$1 \times 1 \times 1 \times 1$$

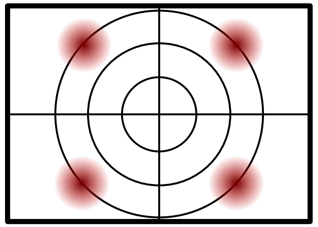
not all simulators
support all combinations

two directions

complex number

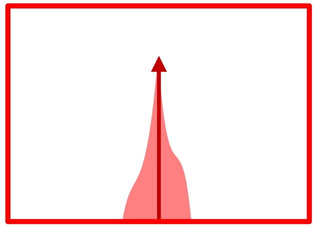


power

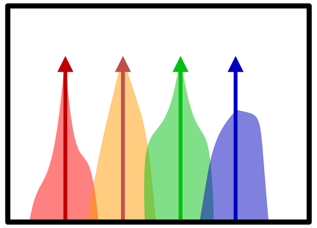


phase

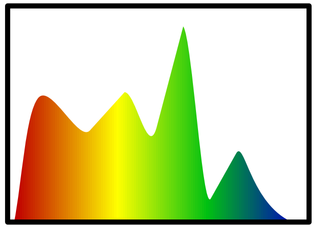
wavelength:
N channels



single

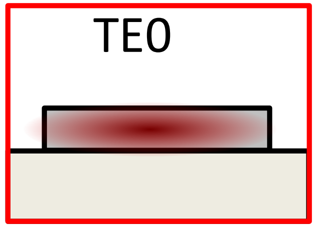


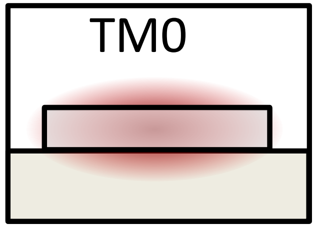
WDM

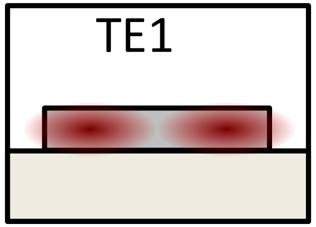


spectrum

mode/polarization:
M modes







GHENT UNIVERSITY imec

Optical Signals: Example

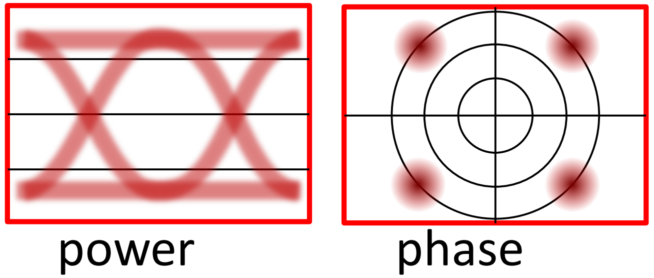
Example: WDM bidirectional link

- two directions
- QPSK modulation: phase
- 32 wavelength channels
- one mode

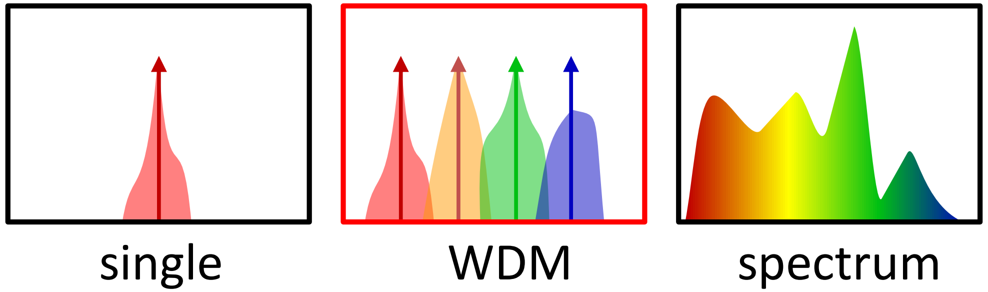
$$2 \times 2 \times 32 \times 1$$

two directions

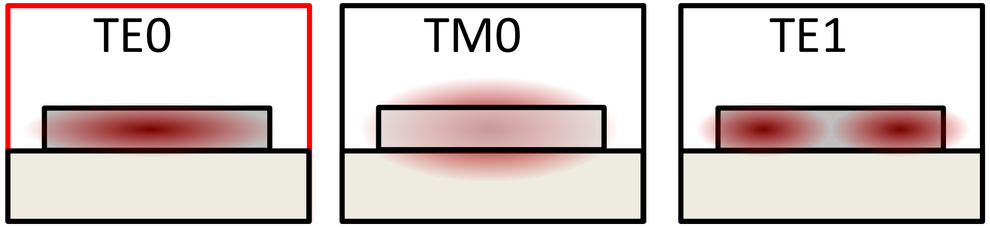
complex number



wavelength:
N channels



mode/polarization:
M modes



Optical Signals: Example

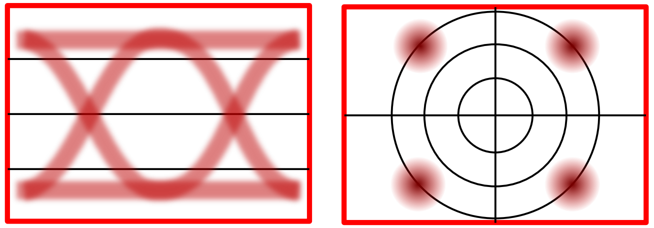
Example: DWDM multimode link

- two directions
- QAM64 modulation: phase
- 512 wavelength channels
- 4 modes

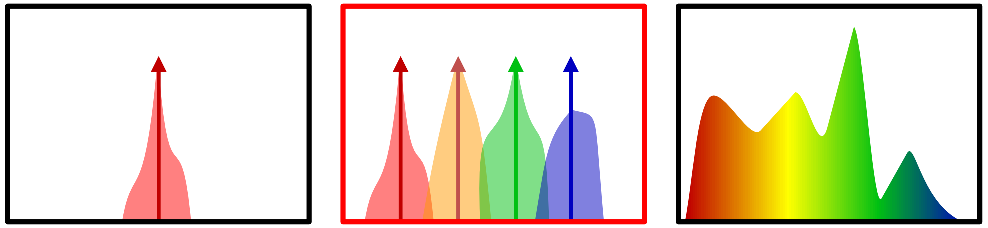
$$2 \times 2 \times 512 \times 4$$

two directions

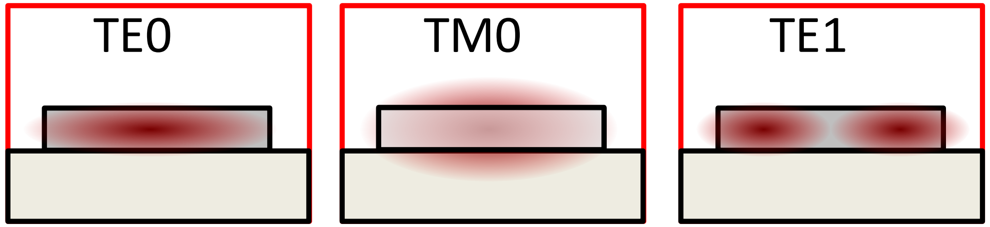
complex number



wavelength:
N channels



mode/polarization:
M modes

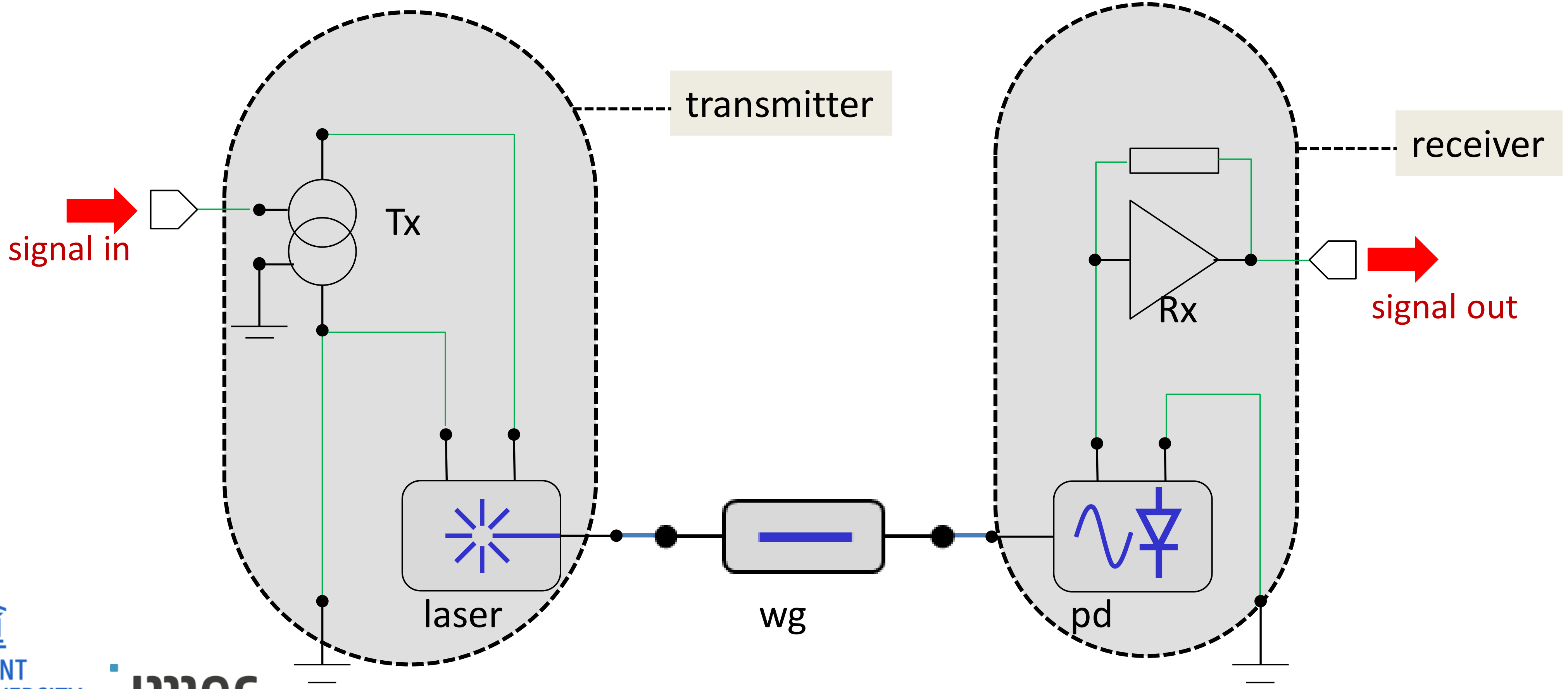


GHENT UNIVERSITY imec

SIMULATING PHOTONICS + ELECTRONICS

Real system: photonics + electronics

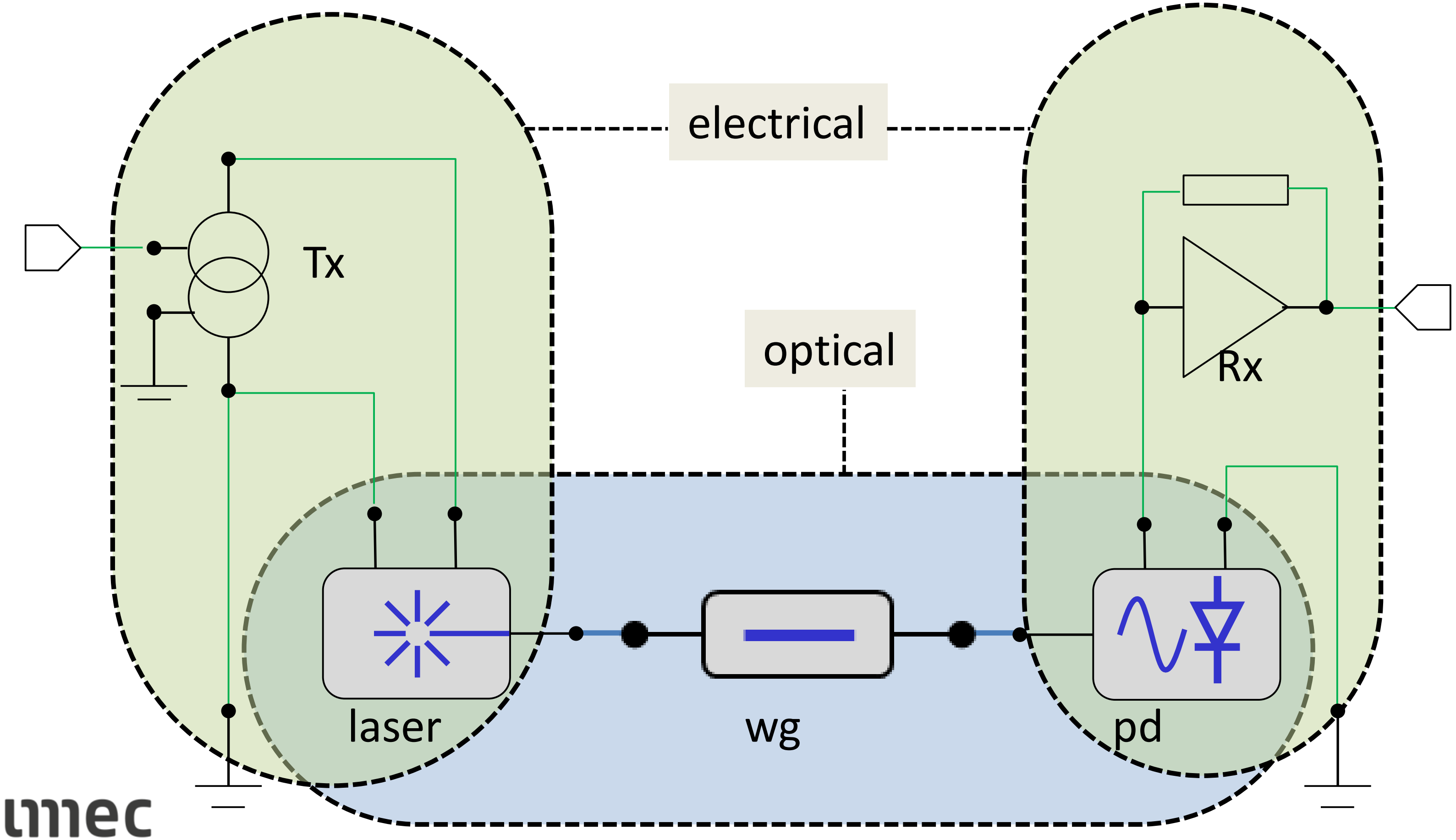
Example: optical link



SIMULATING PHOTONICS + ELECTRONICS

Circuit has optical and electrical parts:

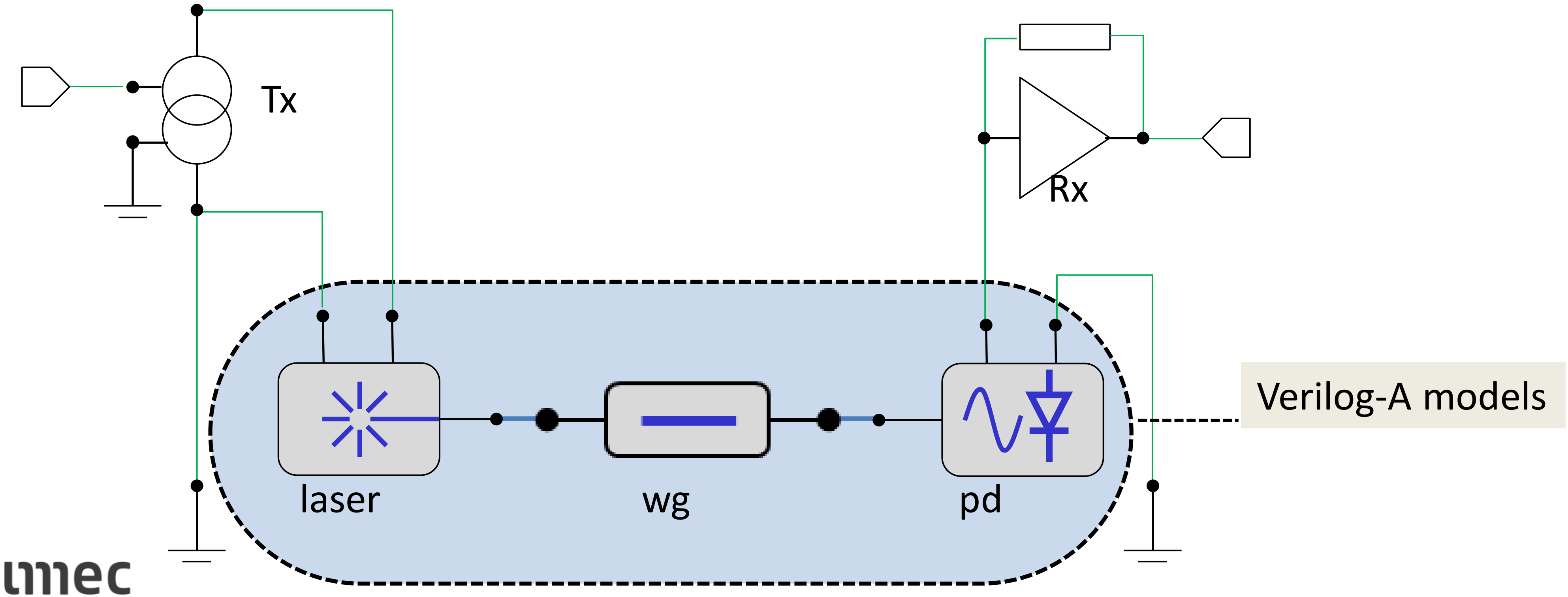
Some components overlap



Simulating Photonics + Electronics

Simulating everything in electrical simulator (SPICE – MNA)

- Use native, verified models for electronics
- Build Verilog-A models for photonics



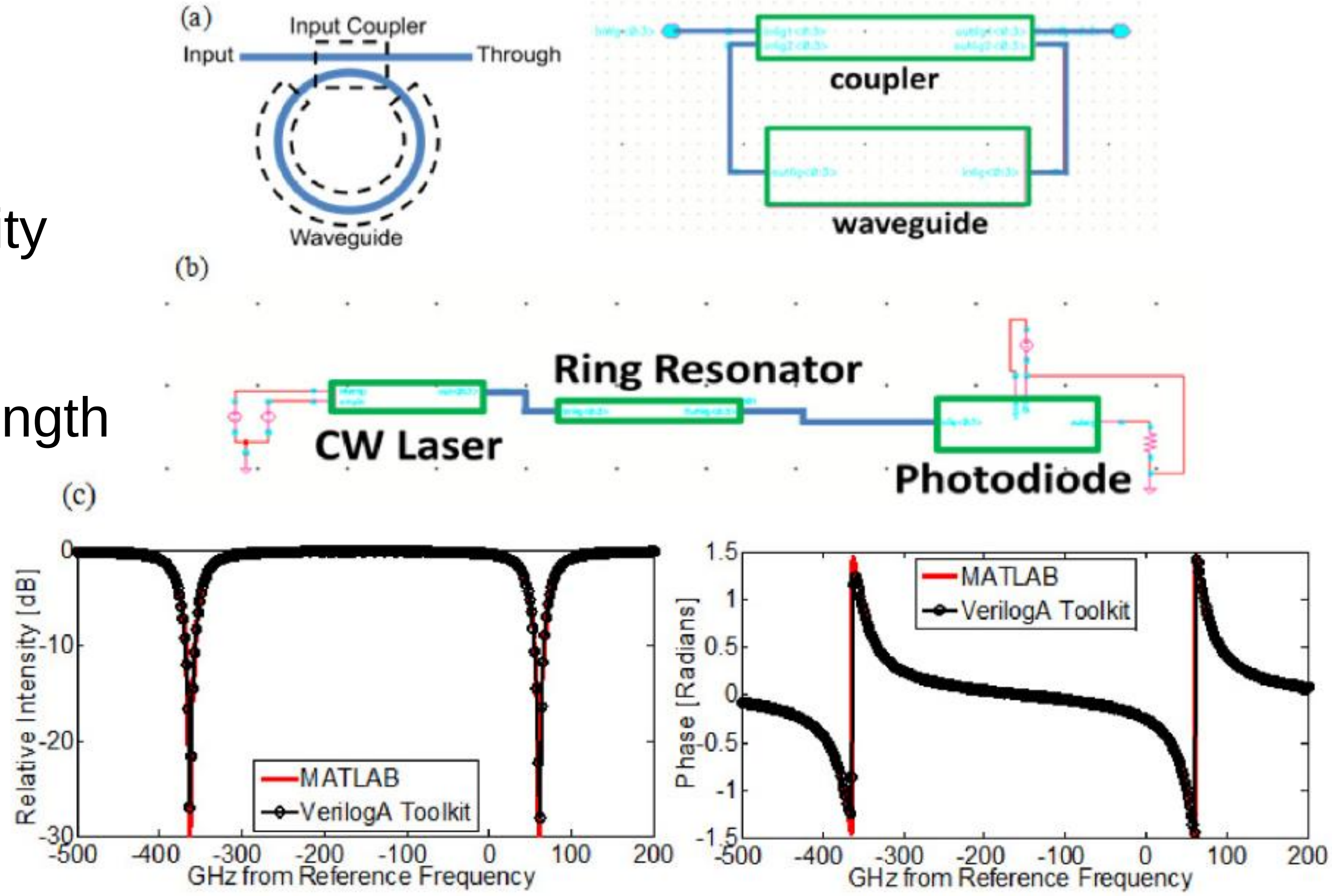
GHENT UNIVERSITY imec

PHOTONICS IN VERILOGA

Encode time signals as 'analytical' signals (complex numbers)

Bus of two lines for bidirectionality

Modulation on an optical wavelength

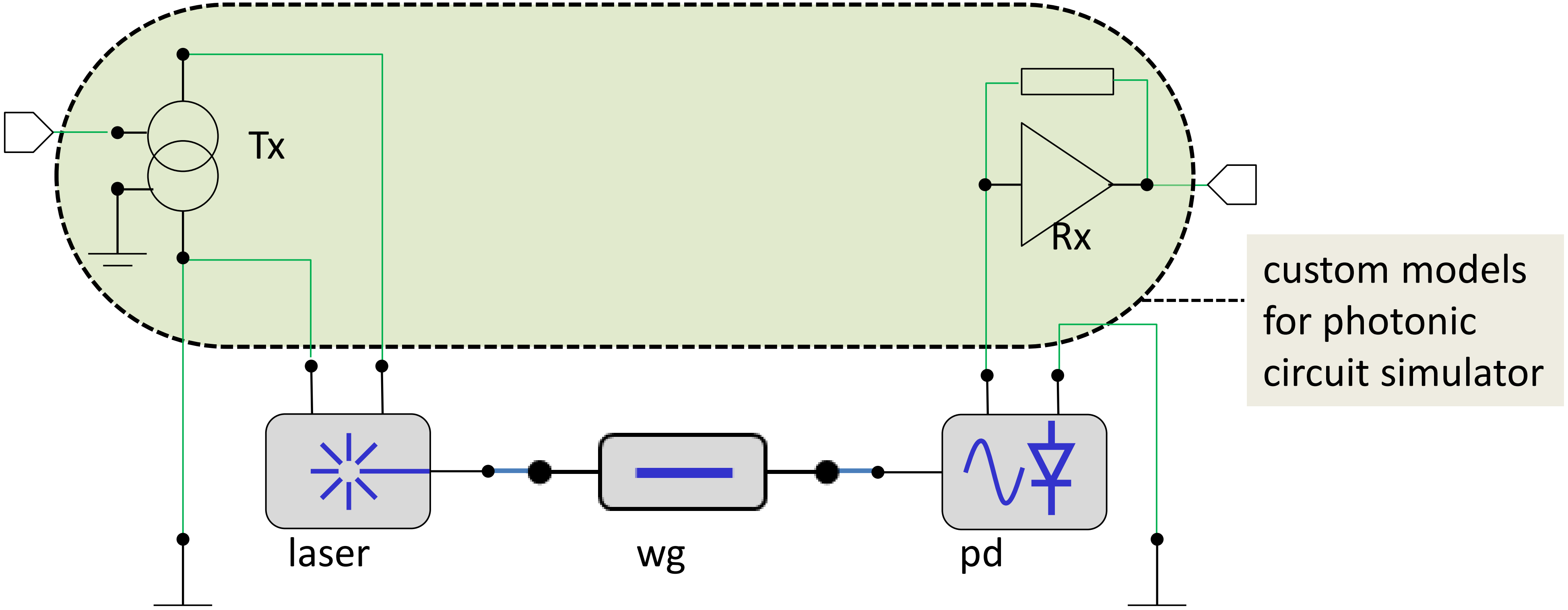


C. Sorace-Agaskar, OpEx 23(21), 2015

SIMULATING PHOTONICS + ELECTRONICS

Simulate everything in a photonics simulator (Interconnect, Caphe, OptSim)

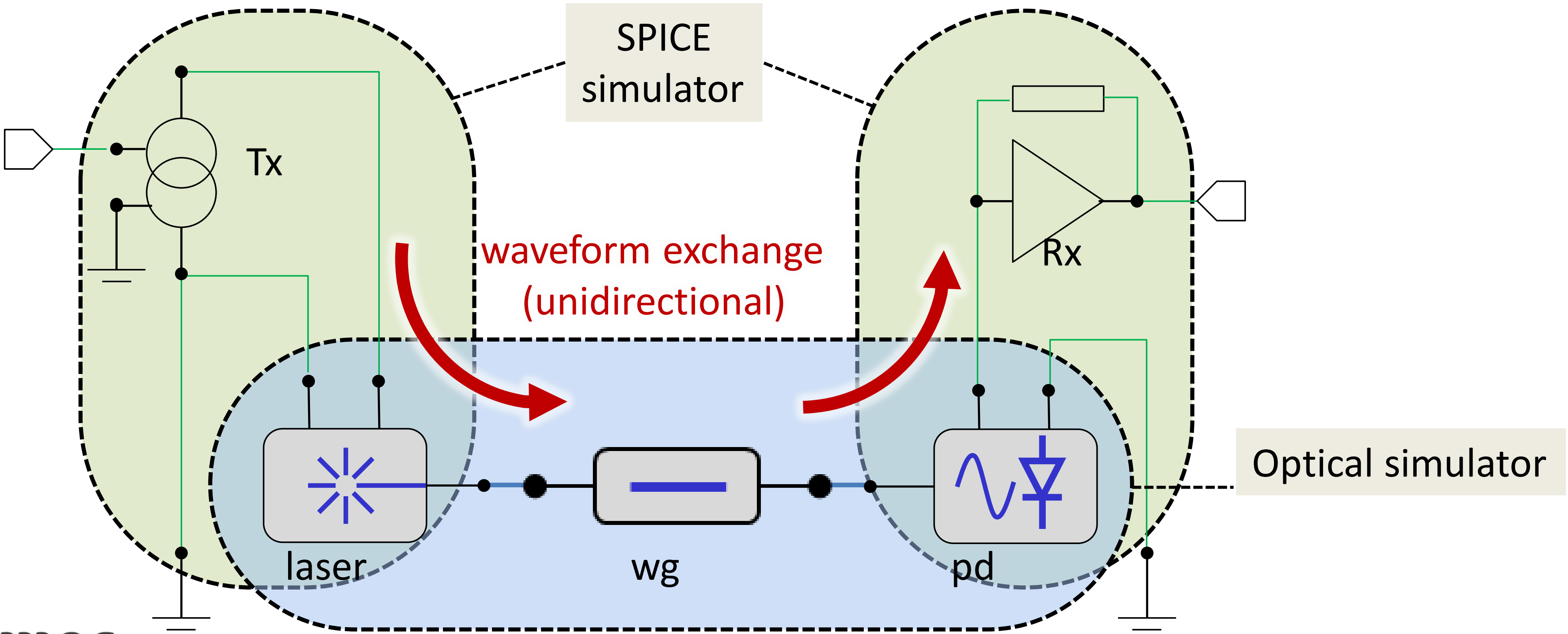
- Optimized models and formalisms for photonics
- Electronics models need to be mapped. No verified fab models



Simulating Photonics + Electronics

Co-simulate with waveform exchange

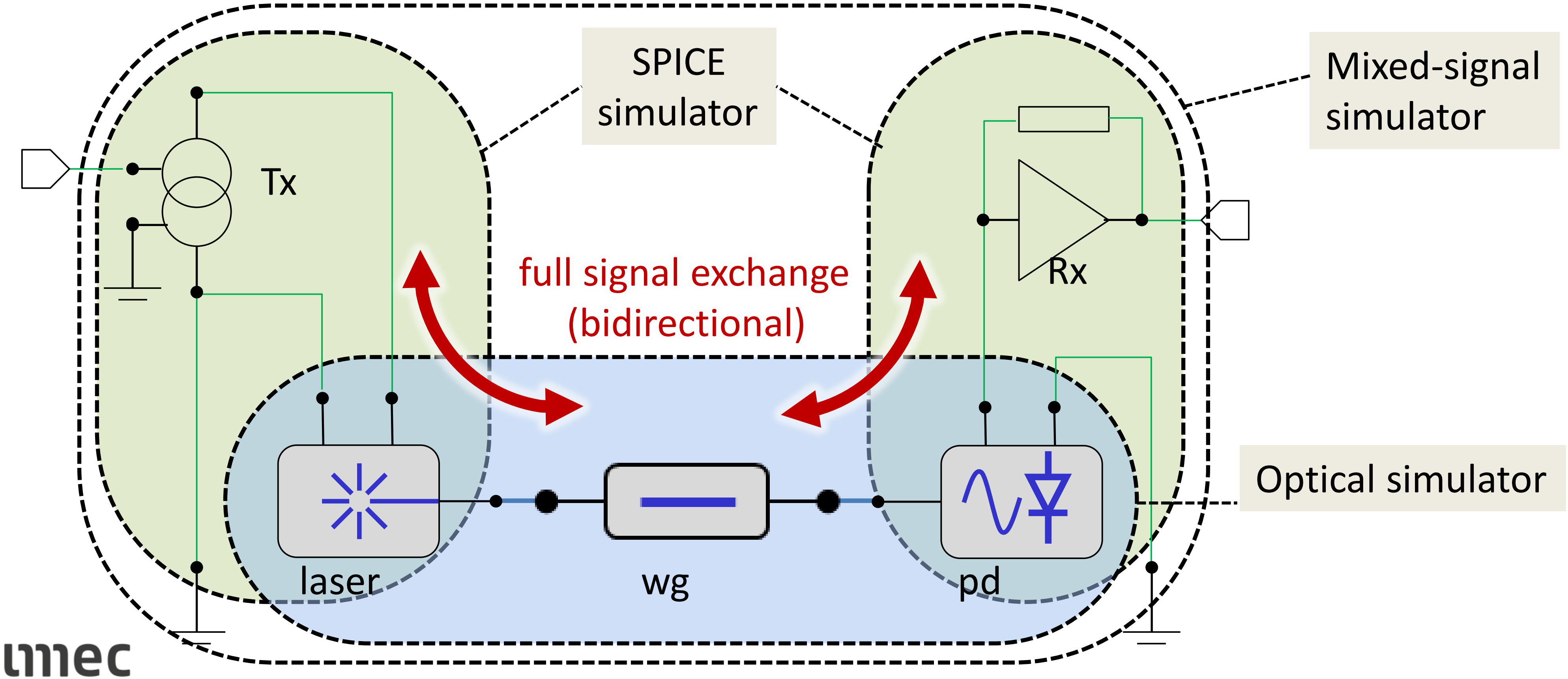
- Photonics and electronics in optimized model, executed sequentially
- Output of one simulation = input of next simulation



SIMULATING PHOTONICS + ELECTRONICS

True cosimulation (photonics and electronics in lockstep)

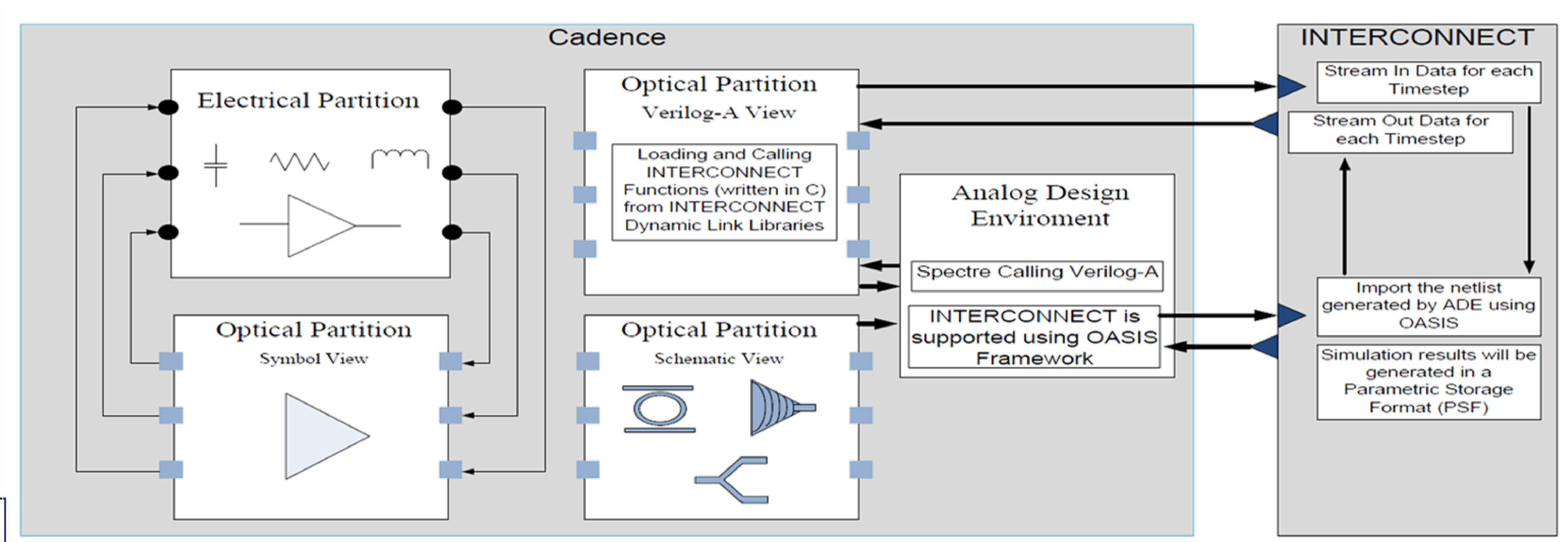
- Both photonic and electronic simulators run in parallel
- Photonic and electronic model exchange data at each step



CO-SIMULATION

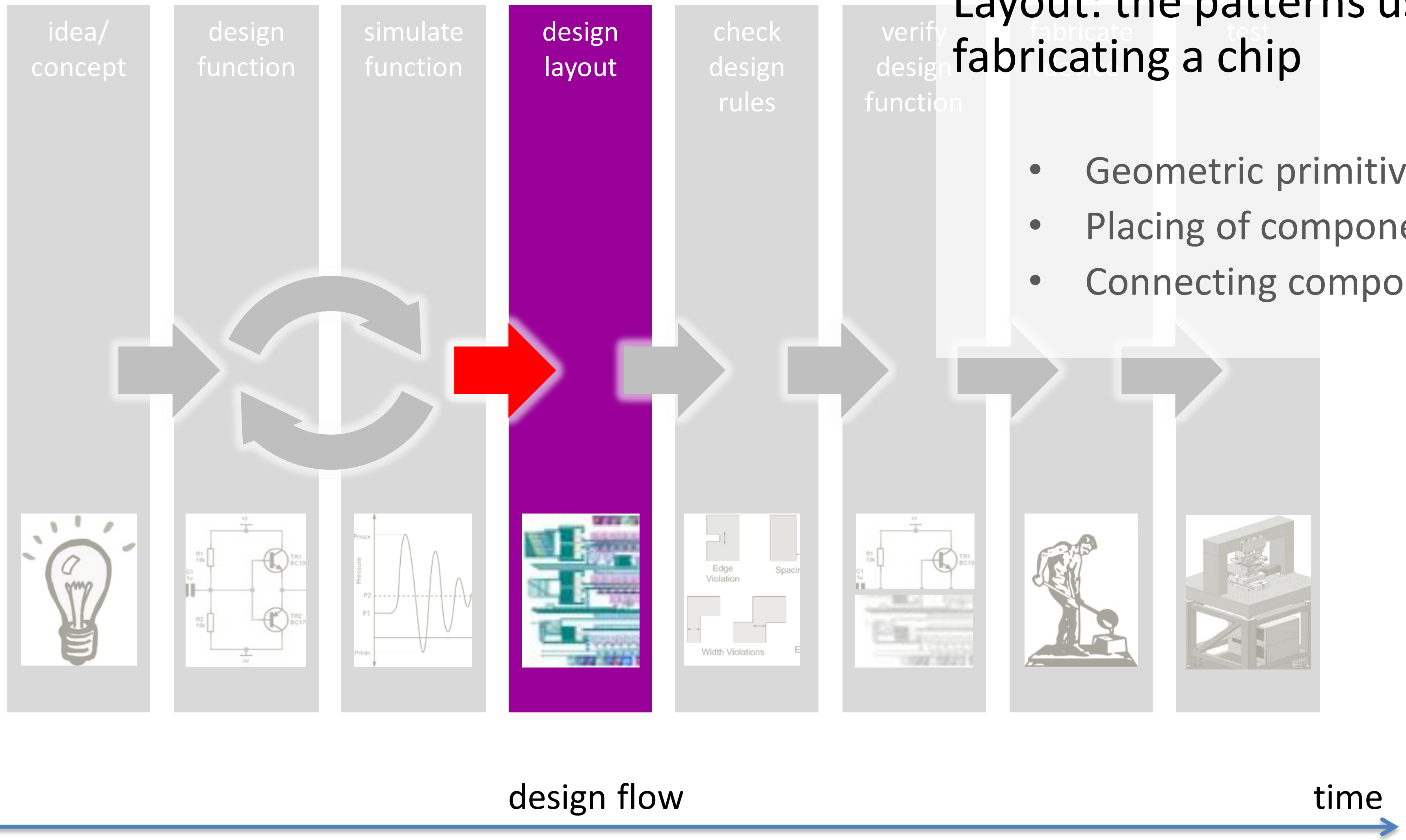
Optical and electrical co-design in Virtuoso Schematic

Photonic simulation in Lumerical Interconnect



A. Farsaei, APC 2016, JTu4A.1

From Function to Layout



Layout: the patterns used for fabricating a chip

- Geometric primitives
- Placing of components
- Connecting components

LAYOUT

Geometric patterns

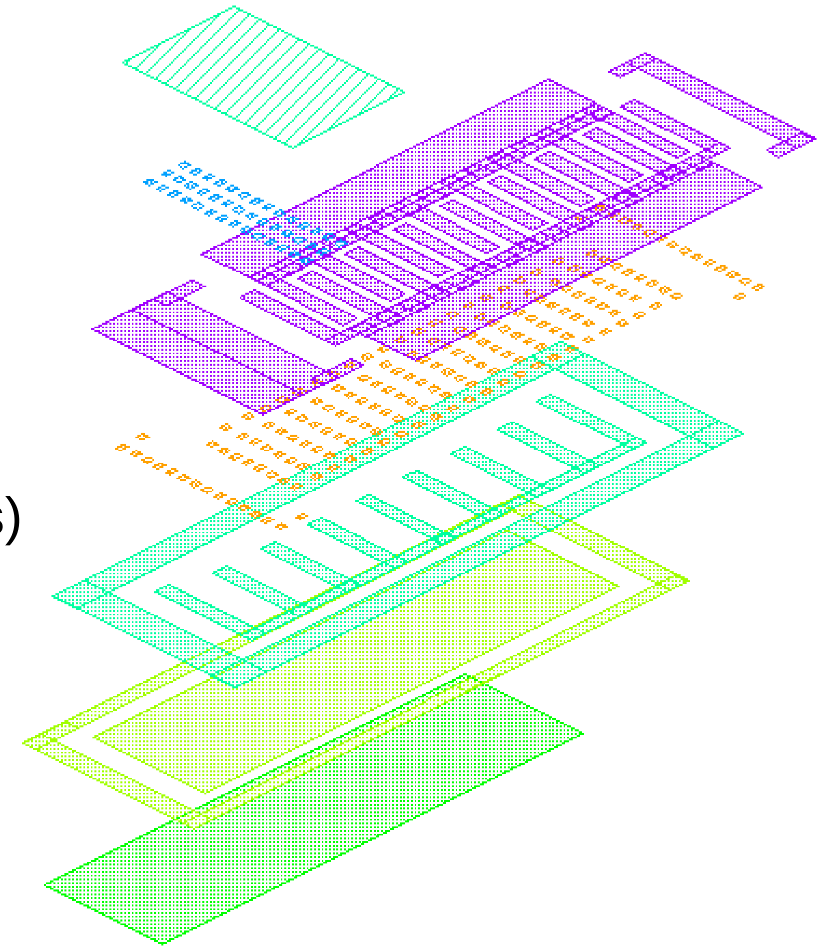
- Originally drawn by hand
- Now drawn by computer
- or programmed using scripts

Different layers

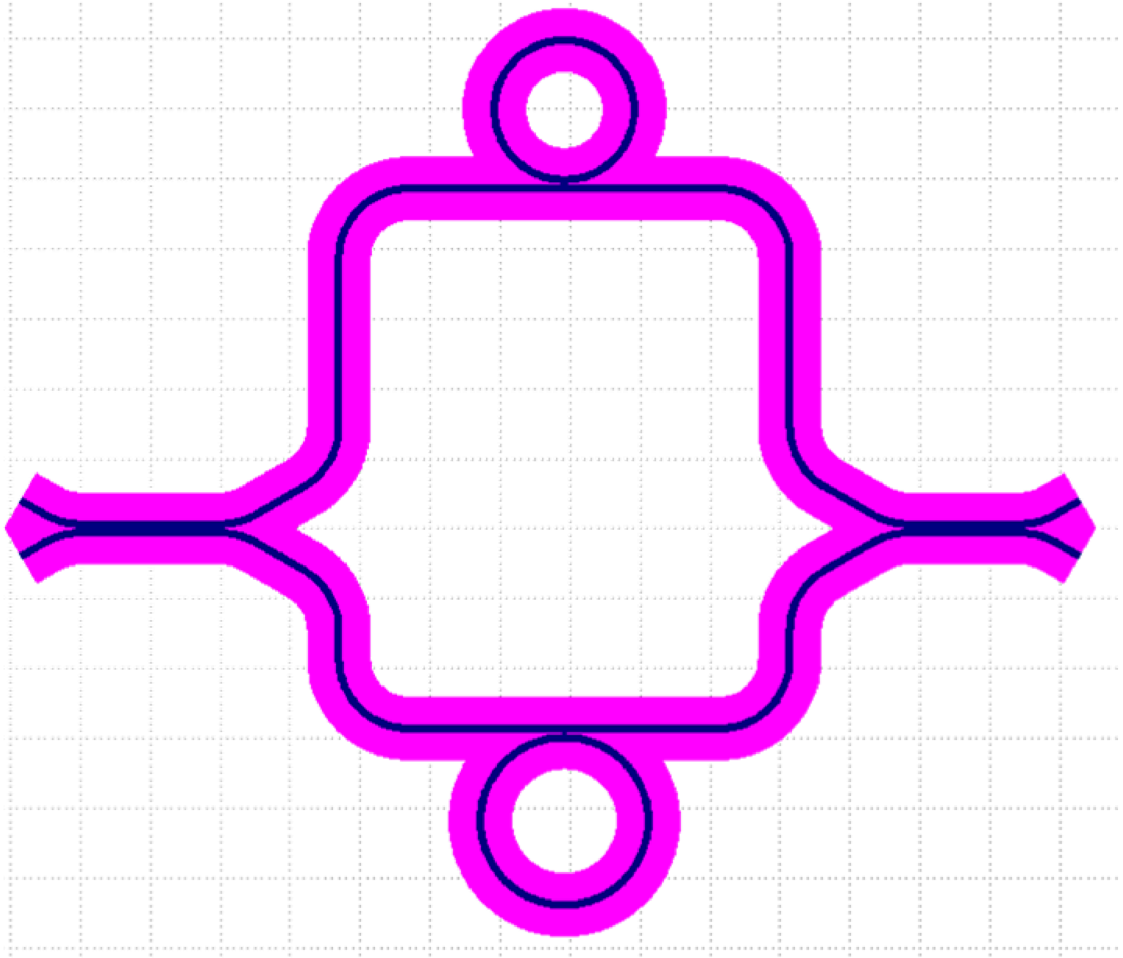
- correspond to process steps: Mask layers
- or to logical operations (e.g. Boolean operations)

Different purposes

- Intent of the drawn shape:
  process, exclusion, annotation, …



LAYOUT: CIRCUITS



Organized in (reusable) Cells

- placement
- transformations

Hierarchy: Cells contain other cells

Routing

- Optical connectivity with waveguides
- Electrical connectivity with metal wiring
- Avoid crossings/shorts/disconnects

PARAMETERIZED CELLS

(Or PCells)

Consists of

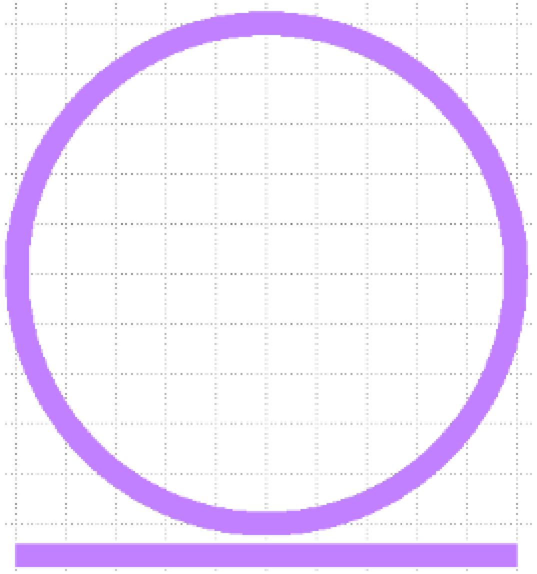
- Parameters
  - that the user can supply
- Evaluators
  - piece(s) of code that generate the content based on the parameters

Layout, model, symbol, netlist, …

Languages:

- Open: Tcl, Python, Ruby
- Proprietary: SKILL, Ample, SPT, …

```
from ipkiss3 import all as i3

class RingResonator(i3.PCell):

    class Layout(i3.LayoutView):

        ring_radius = i3.PositiveNumberProperty(default=20.0)
        wg_width = i3.PositiveNumberProperty(default=0.45)
        coupler_gap = i3.PositiveNumberProperty(default=0.3)

        def _generate_elements(self, elems):
            r = self.ring_radius
            g = self.coupler_gap
            w = self.wg_width

            elems += i3.CirclePath(layer=i3.Layer(2),
                                   radius=r,
                                   line_width=w)
            elems += i3.Line(layer=Layer(2),
                             begin_coord=(-r, -r-w-g),
                             end_coord=(+r, -r-w-g),
                             line_width=w)
            return elems
```



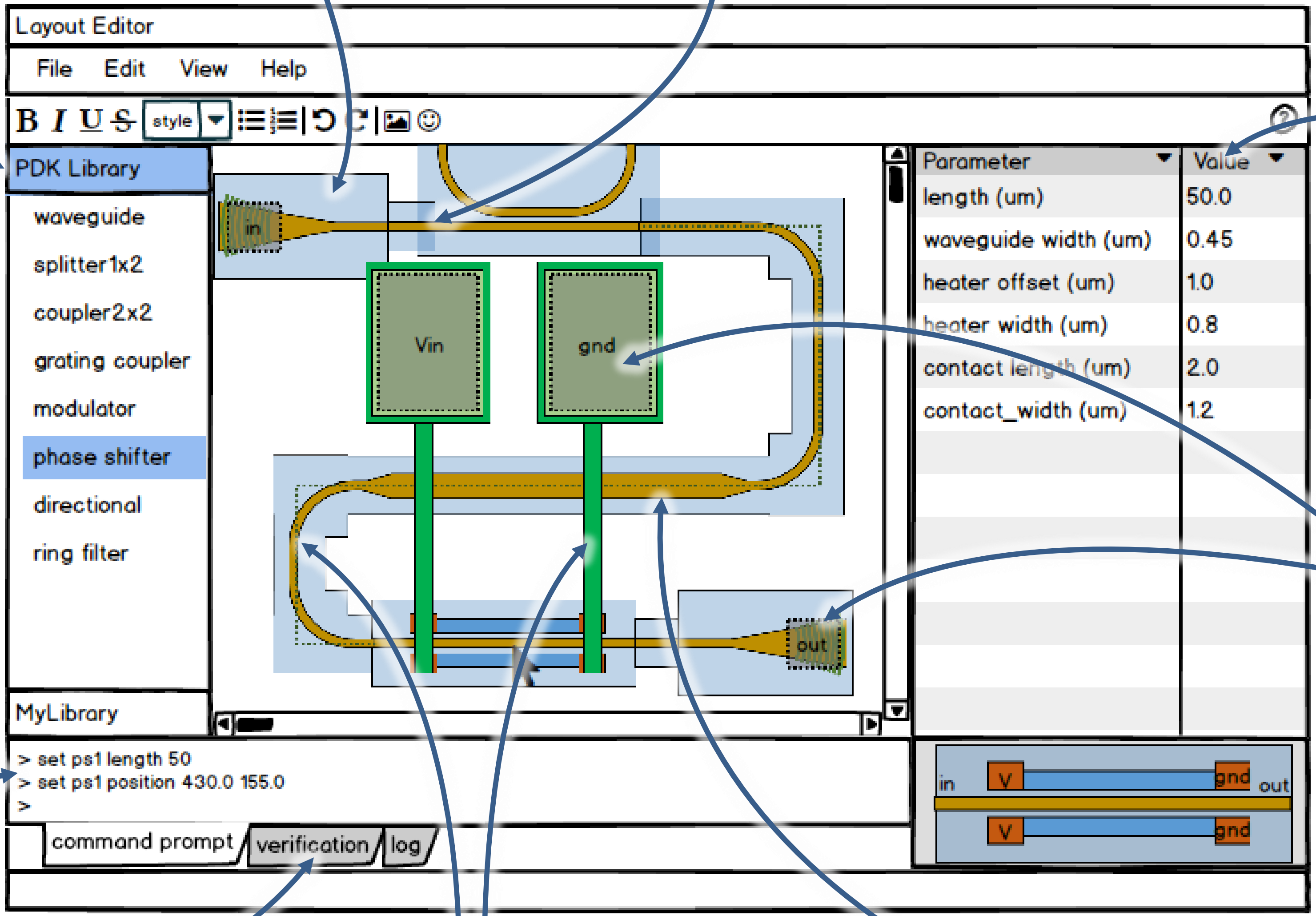
Layout Editors

drag and dropping components

alignment and snapping at waveguide ports

component libraries

parametrization



optical and electrical pins

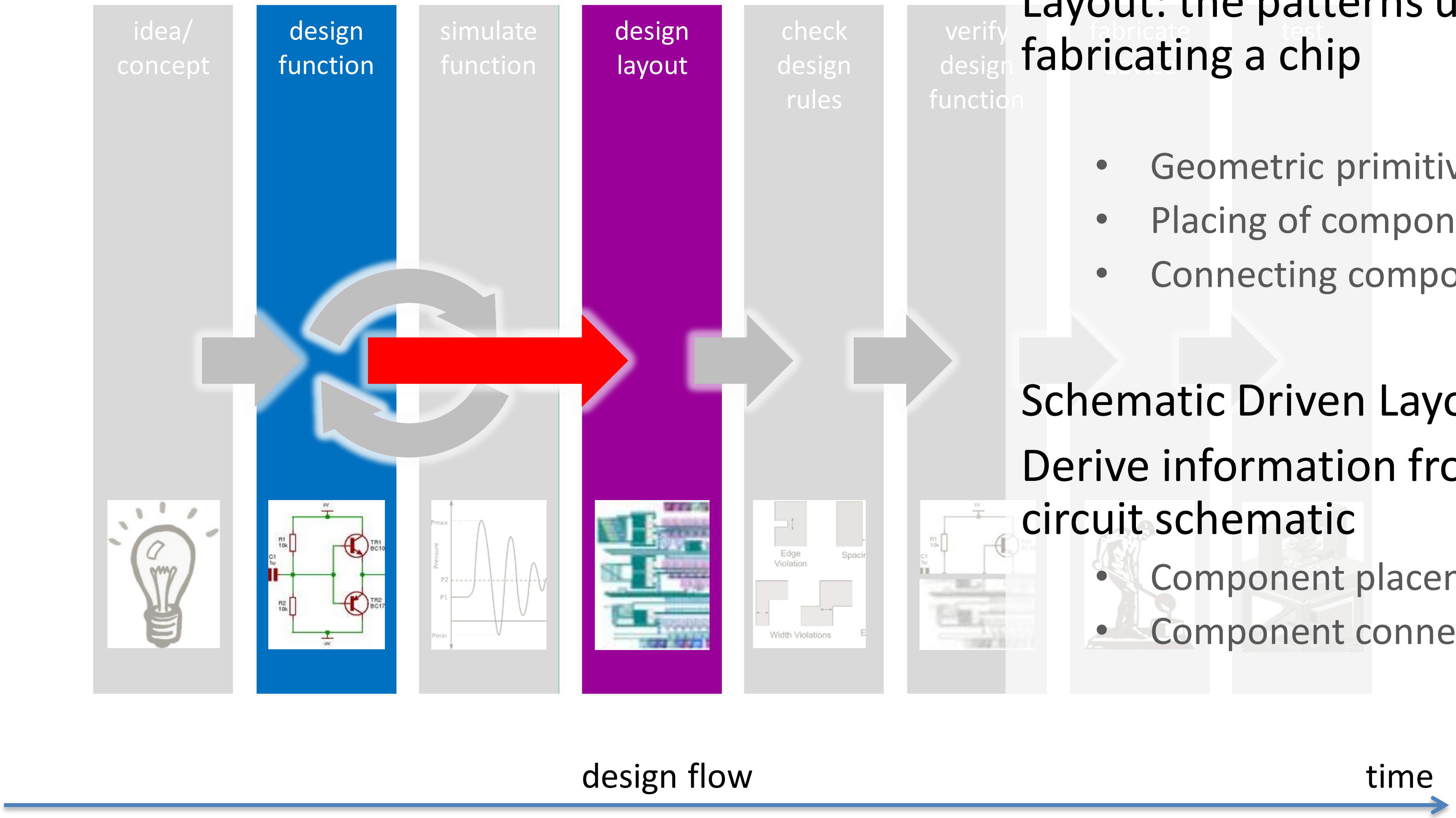
scriptability

interface to verification (DRC and LVS)

routing of waveguides and electrical wires

smart waveguide cells with automatic bend radius and flaring in long segments

SCHEMATIC DRIVEN LAYOUT (SDL)



Layout: the patterns used for fabricating a chip

- Geometric primitives
- Placing of components
- Connecting component

Schematic Driven Layout:

Derive information from circuit schematic

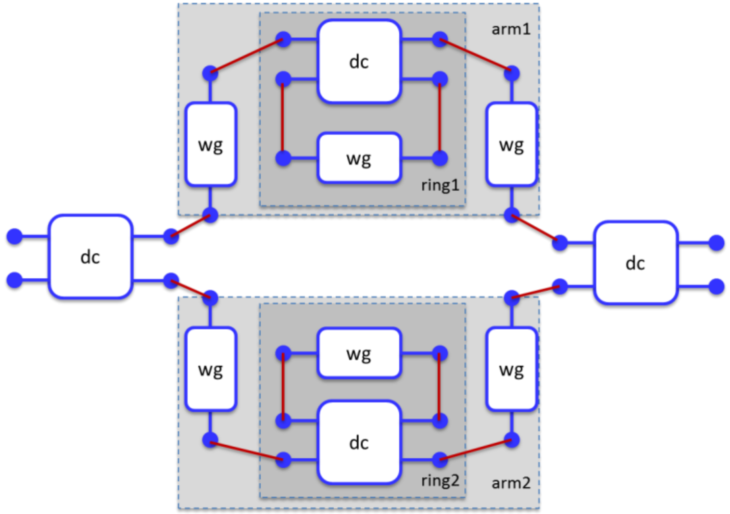
- Component placement
- Component connectivity

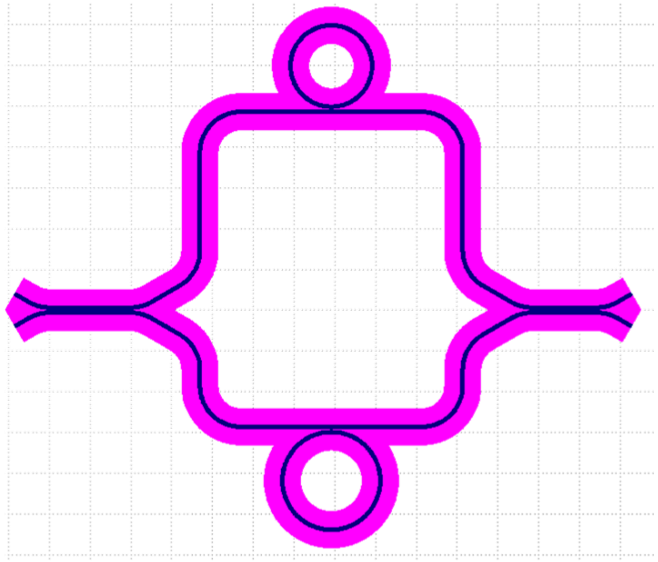
GHENT UNIVERSITY imec

Schematic Driven Layout (SDL)

Derive the physical layout from the schematic

- Generate the Layout (P)Cells
- Place the Layout Cells
- Connect the layout cells together





Not trivial to fully automate

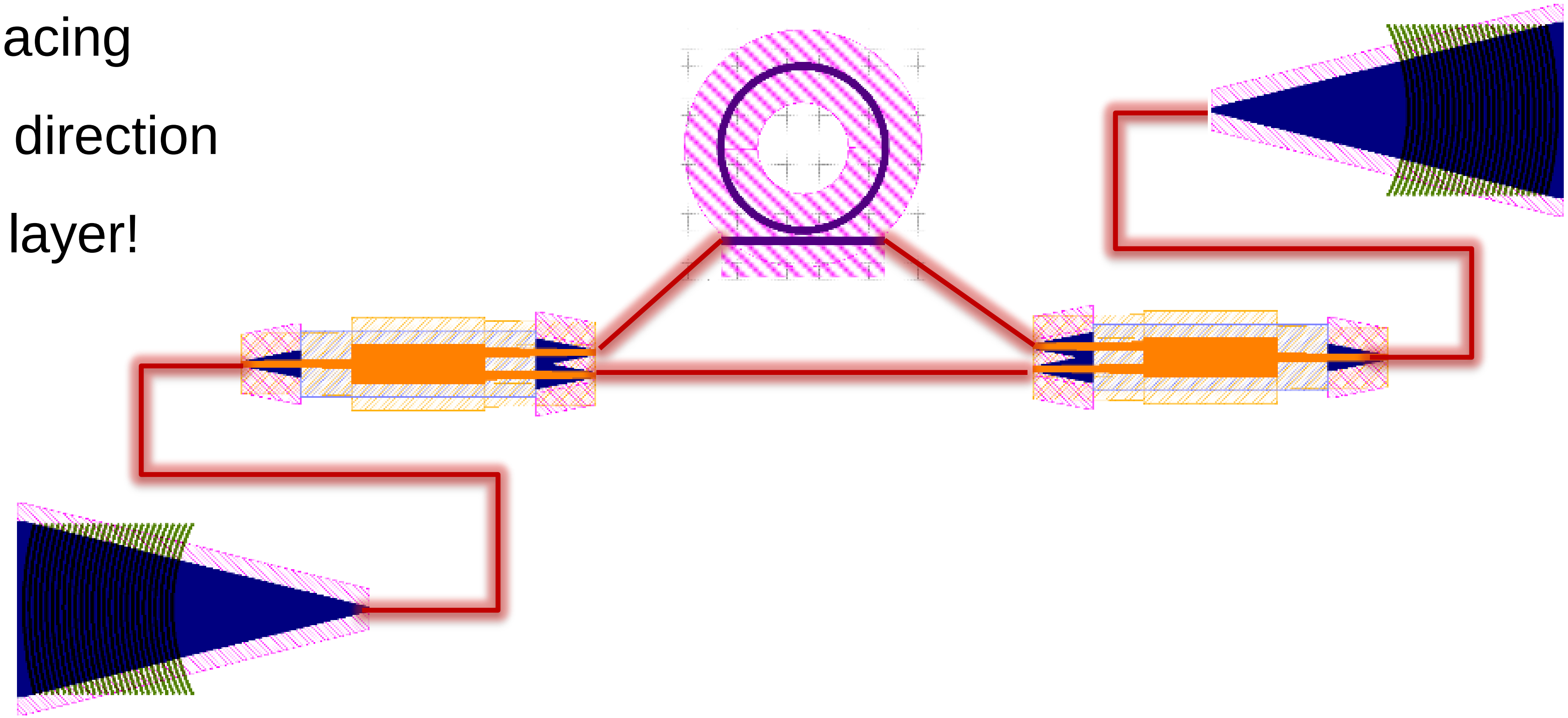
- What is the optimal placement?
- Is the topology possible?
- Constraints for length matching?
- On which layer to route?
- Waveguide bends and crossings?

Combination of manual + auto

PLACEMENT AND ROUTING

Photonic-specific constraints

- ‘optical length’ and phase control
- minimal bend radius
- waveguide spacing
- matching port direction
- single routing layer!

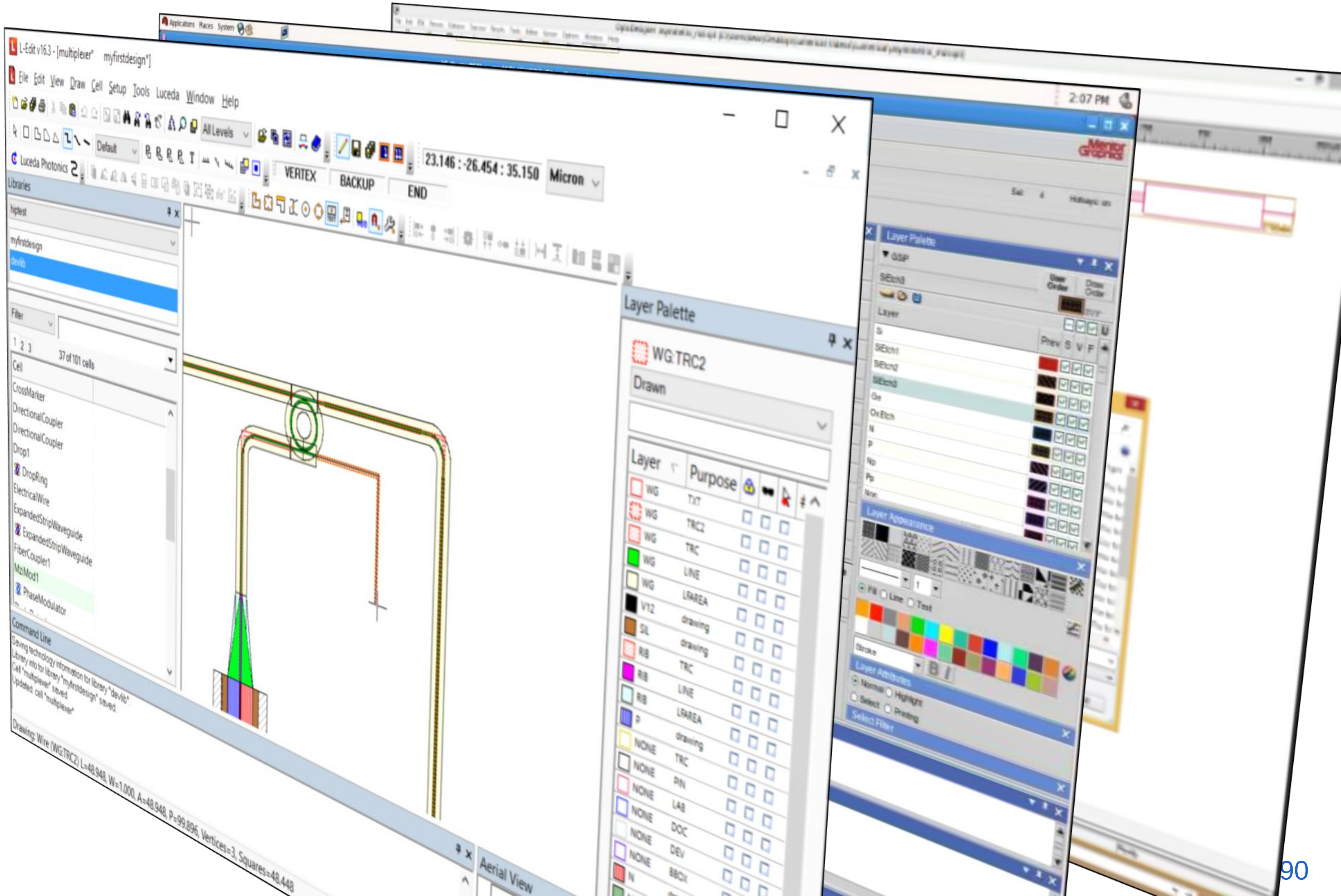


GHENT UNIVERSITY imec

PHOTONIC SDL TOOLS ARE EMERGING

Pure photonics
or based on EDA tools

- define connections
- place components
- route waveguides

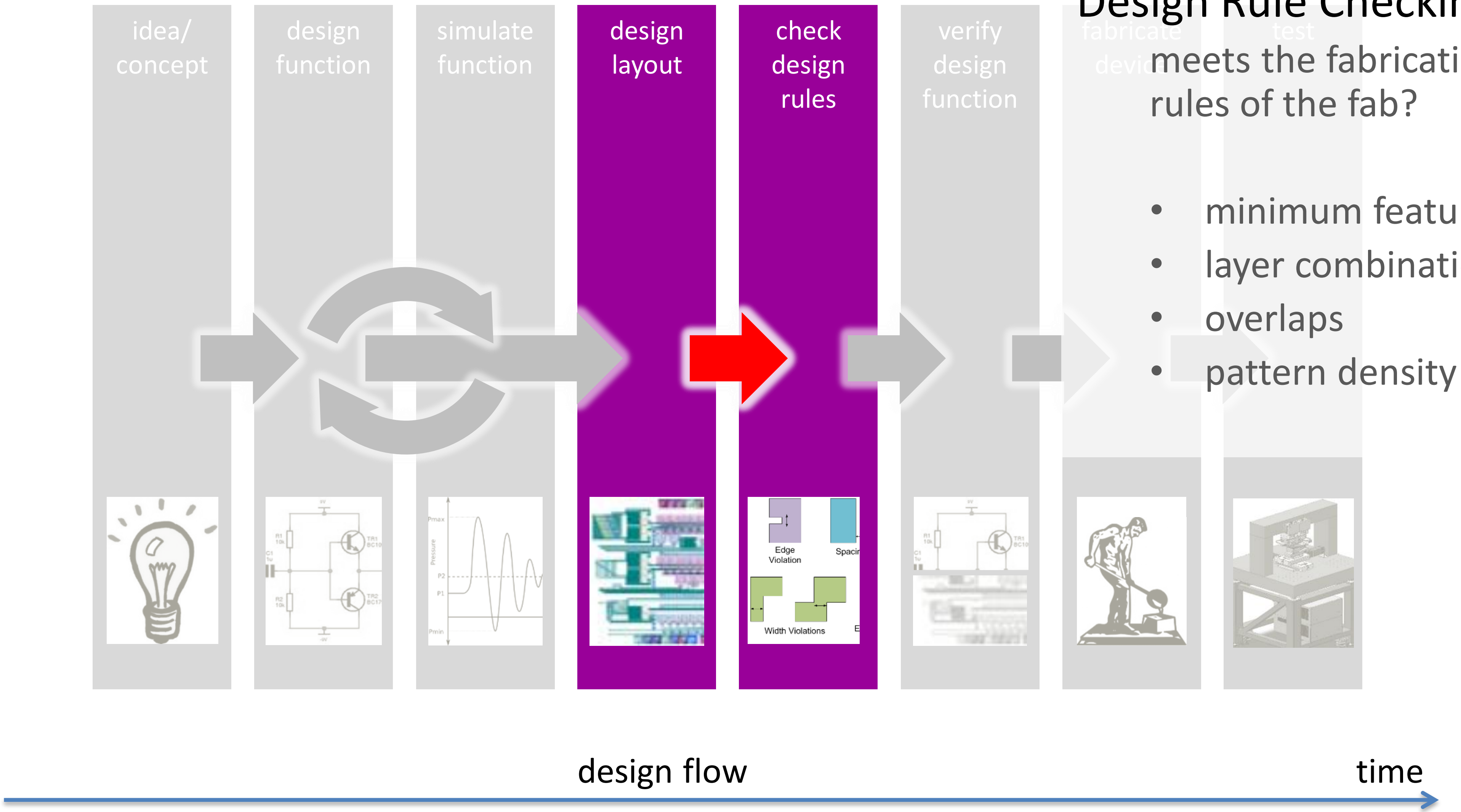


Luceda, Phoenix,

Mentor Graphics,

Cadence …

Is the layout valid?



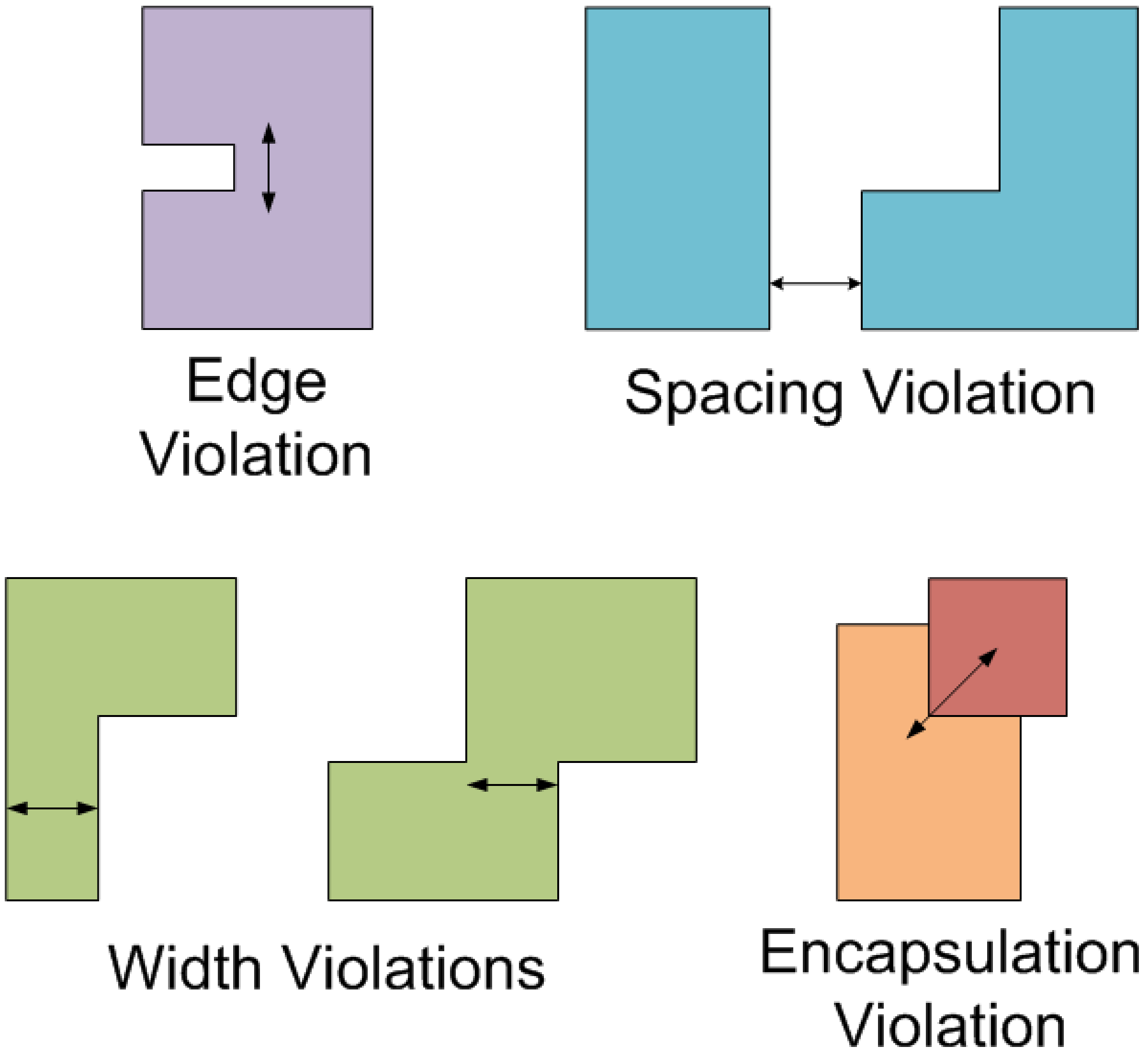
Design Rule Checking

meets the fabrication rules of the fab?

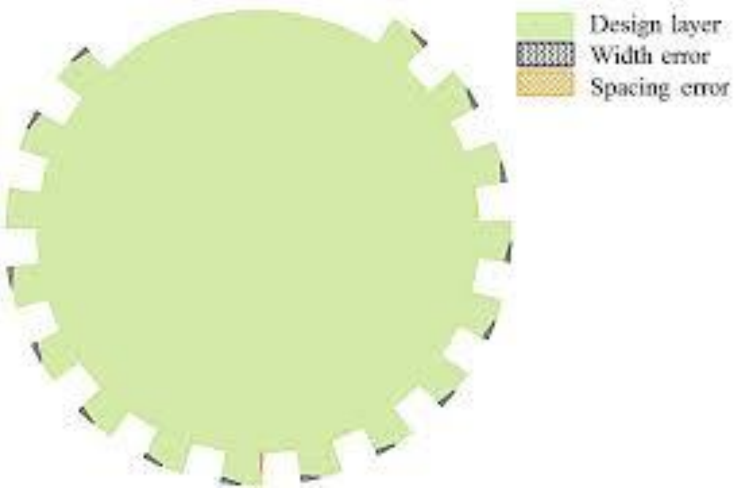
- minimum features
- layer combinations
- overlaps
- pattern density

GHENT UNIVERSITY imec

DESIGN RULE VIOLATIONS: EXAMPLES



PHOTONIC PROBLEMS WITH DRC?



DRC techniques were designed for electronics:

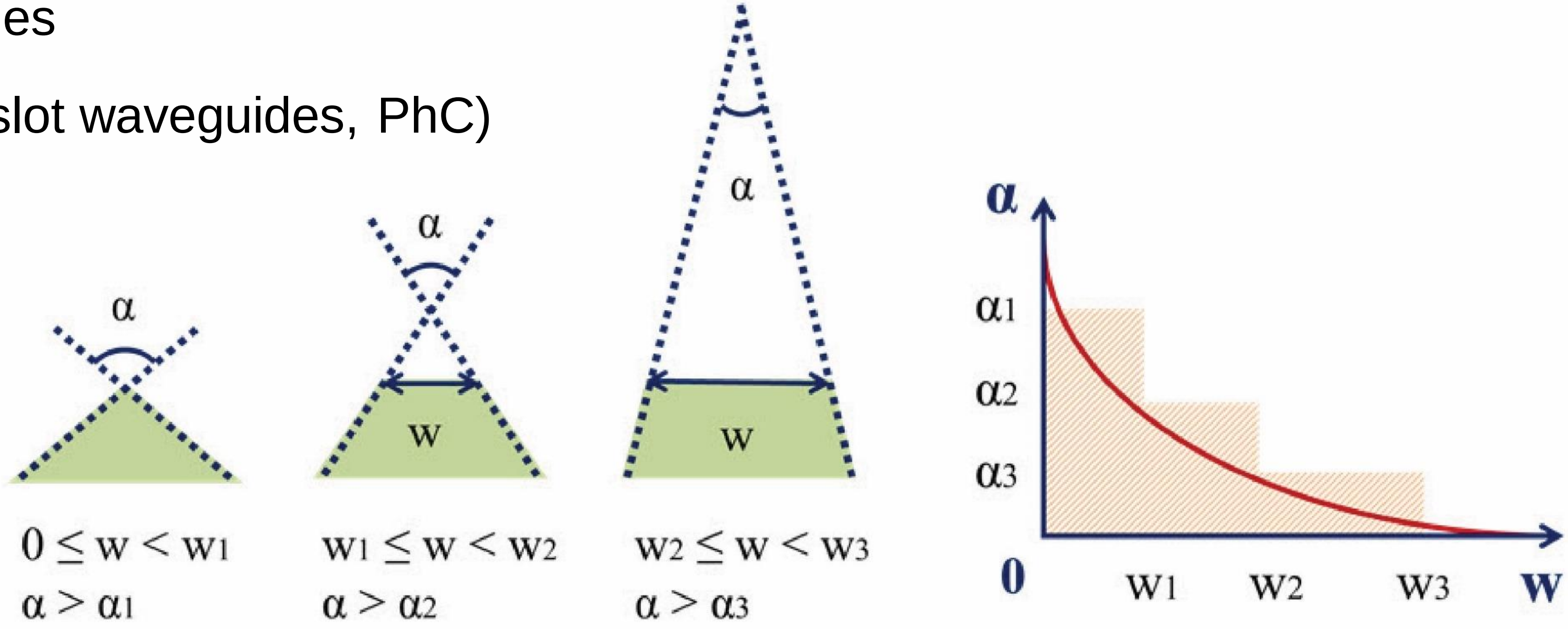
90-degree angles…

Silicon Photonics:

- All-angle waveguides – discretized…
- Nanometer scale sensitivities
- Arbitrary geometries (e.g. slot waveguides, PhC)

What is bad?

What is intentional?



R. Cao, VSLI-SoC 2014

GHENT UNIVERSITY imec

PATTERN DENSITIES

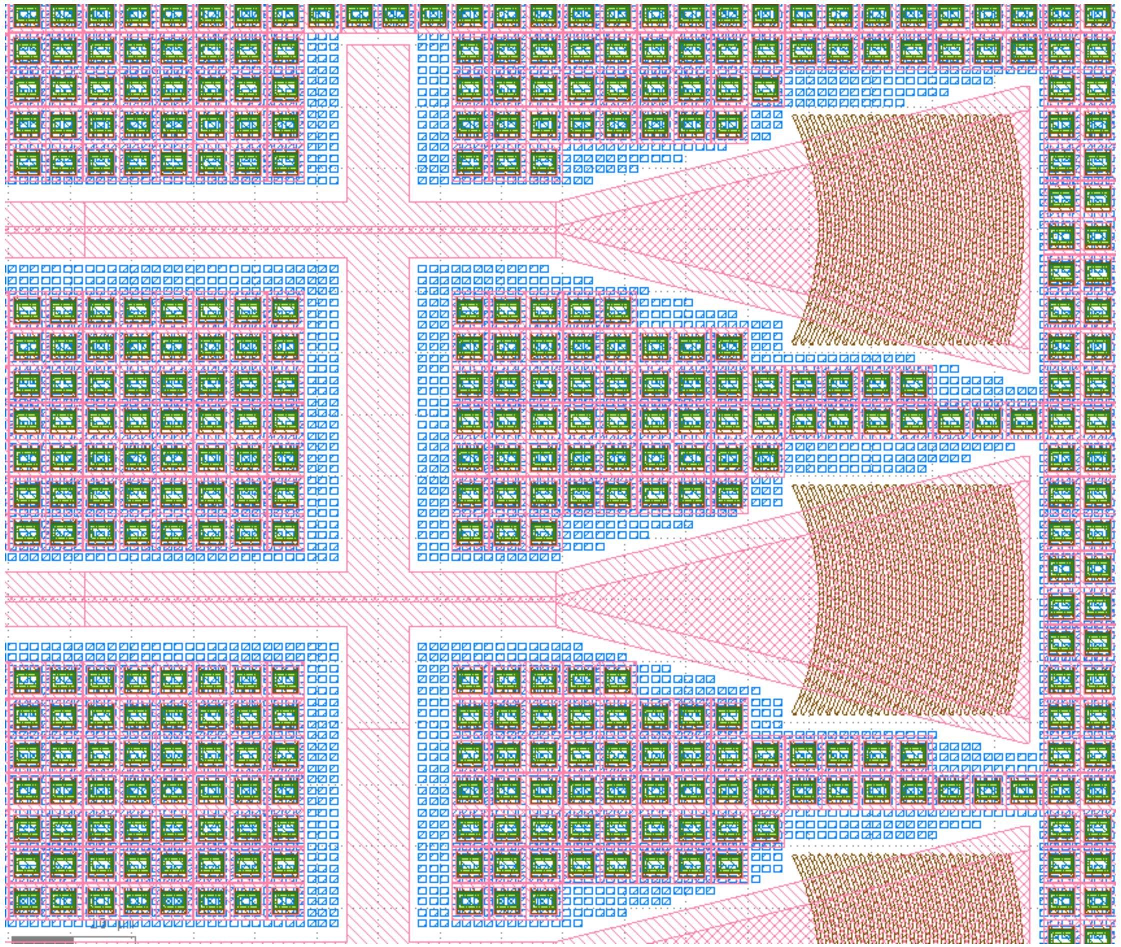
Pattern density must be sufficiently uniform

- Etch rate control
- Avoid CMP dishing

Tiles are added

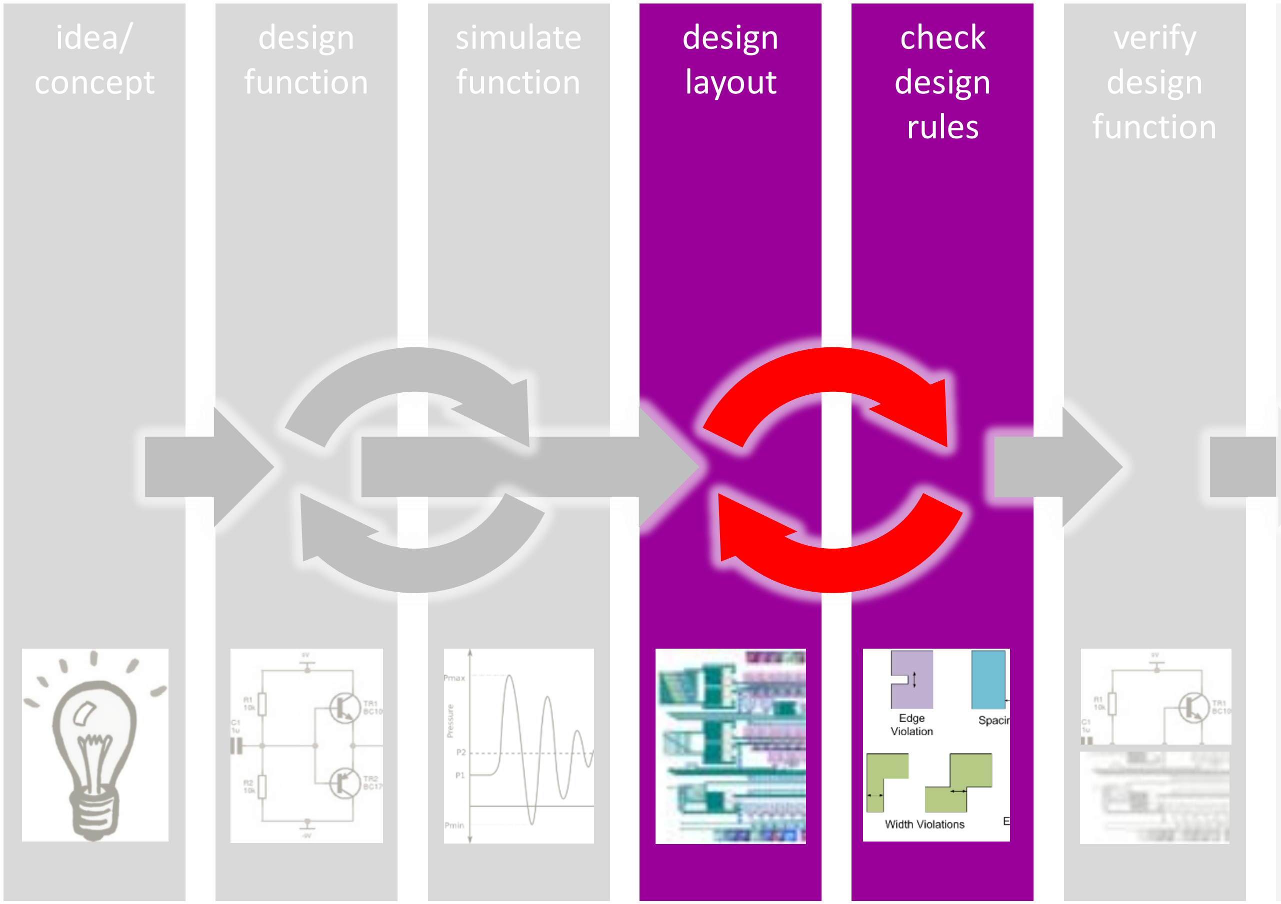
There must be sufficient room to add tiles

- Slab areas (AWG)
- Dense waveguide arrays
- …



GHENT UNIVERSITY · imec

IS THE LAYOUT VALID?



Design Rule Checking

meets the fabrication rules of the fab?

- minimum features
- layer combinations
- overlaps
- pattern density

An iterative process

design flow

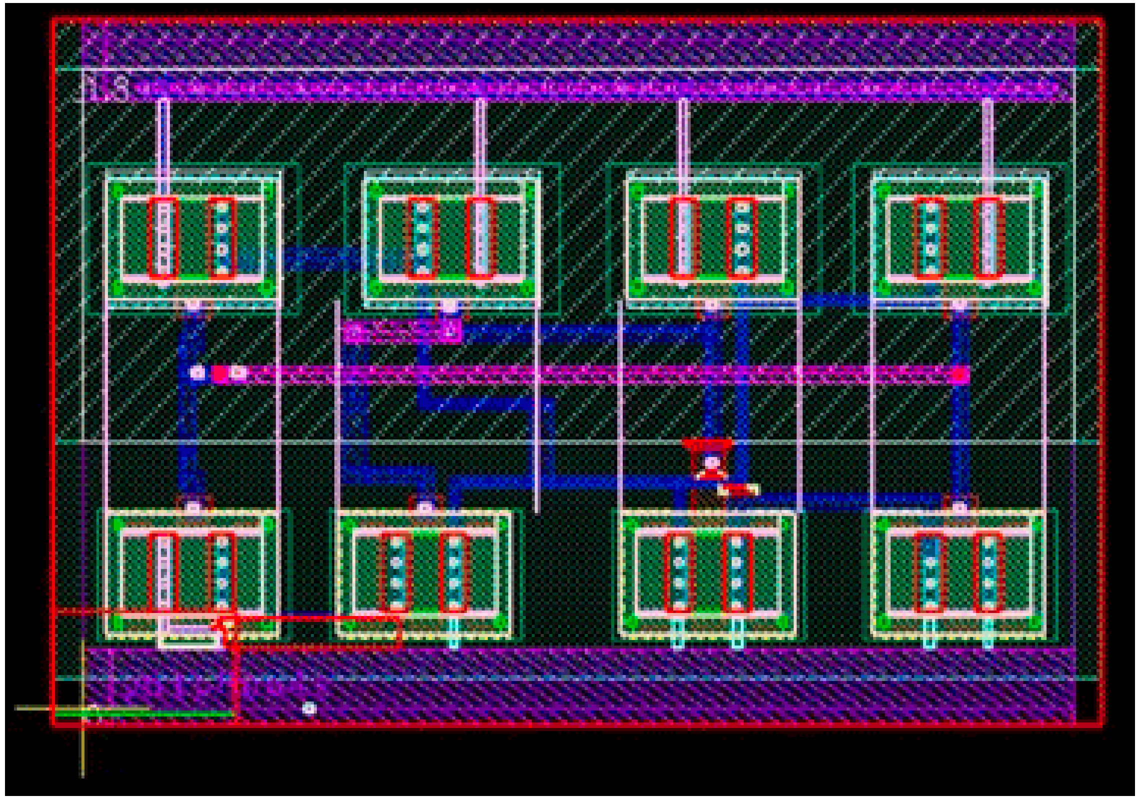
time

REAL-TIME DRC

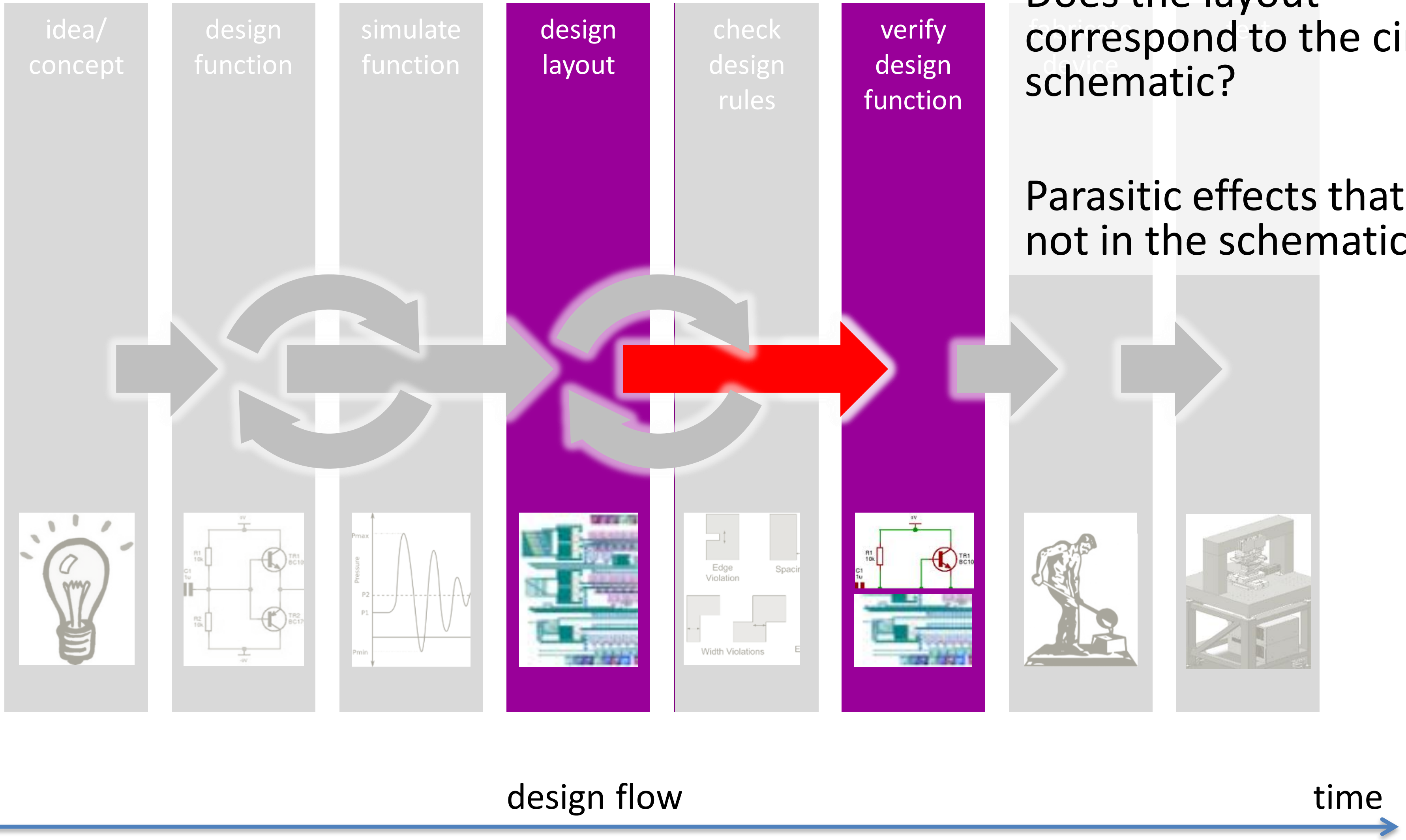
Layout is checked on DRC errors as it is being generated

Real-time feedback in editor

Much faster to a DRC-clean design



FUNCTIONAL VERIFICATION



Does the layout correspond to the circuit schematic?

Parasitic effects that were not in the schematic

Functional Verification: Layout versus Schematic

Check Connectivity

Are the correct components placed?

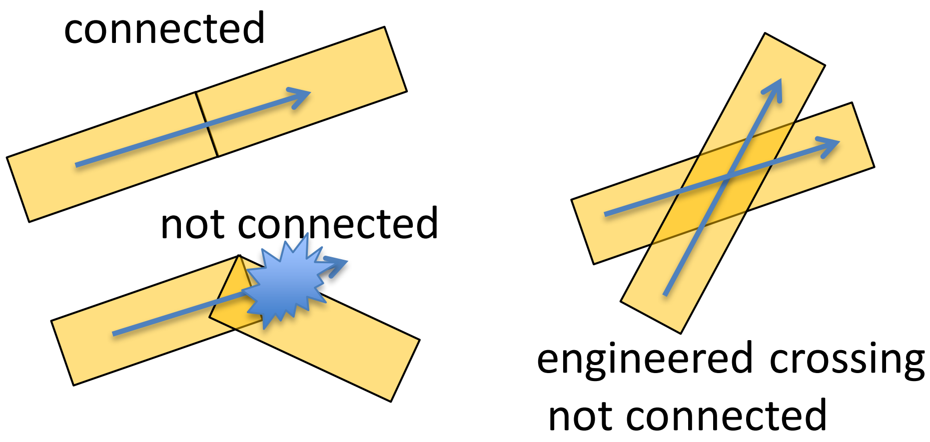
Are they properly connected?

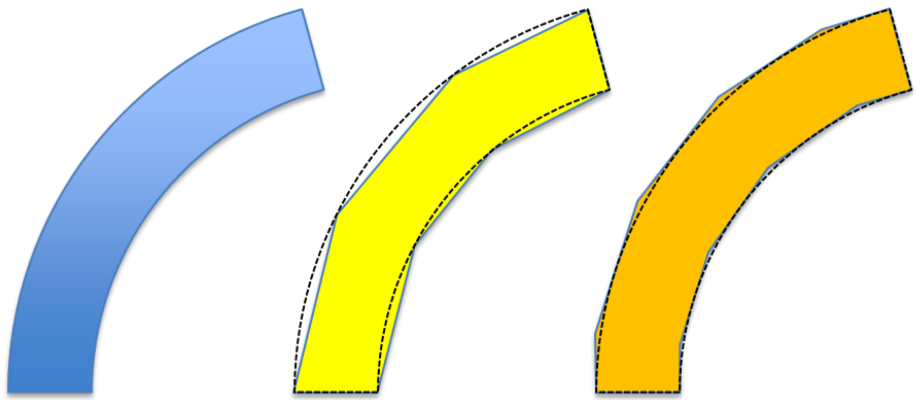
Check functionality

Did we use the right parameters?

Does the layout perform the correct function?

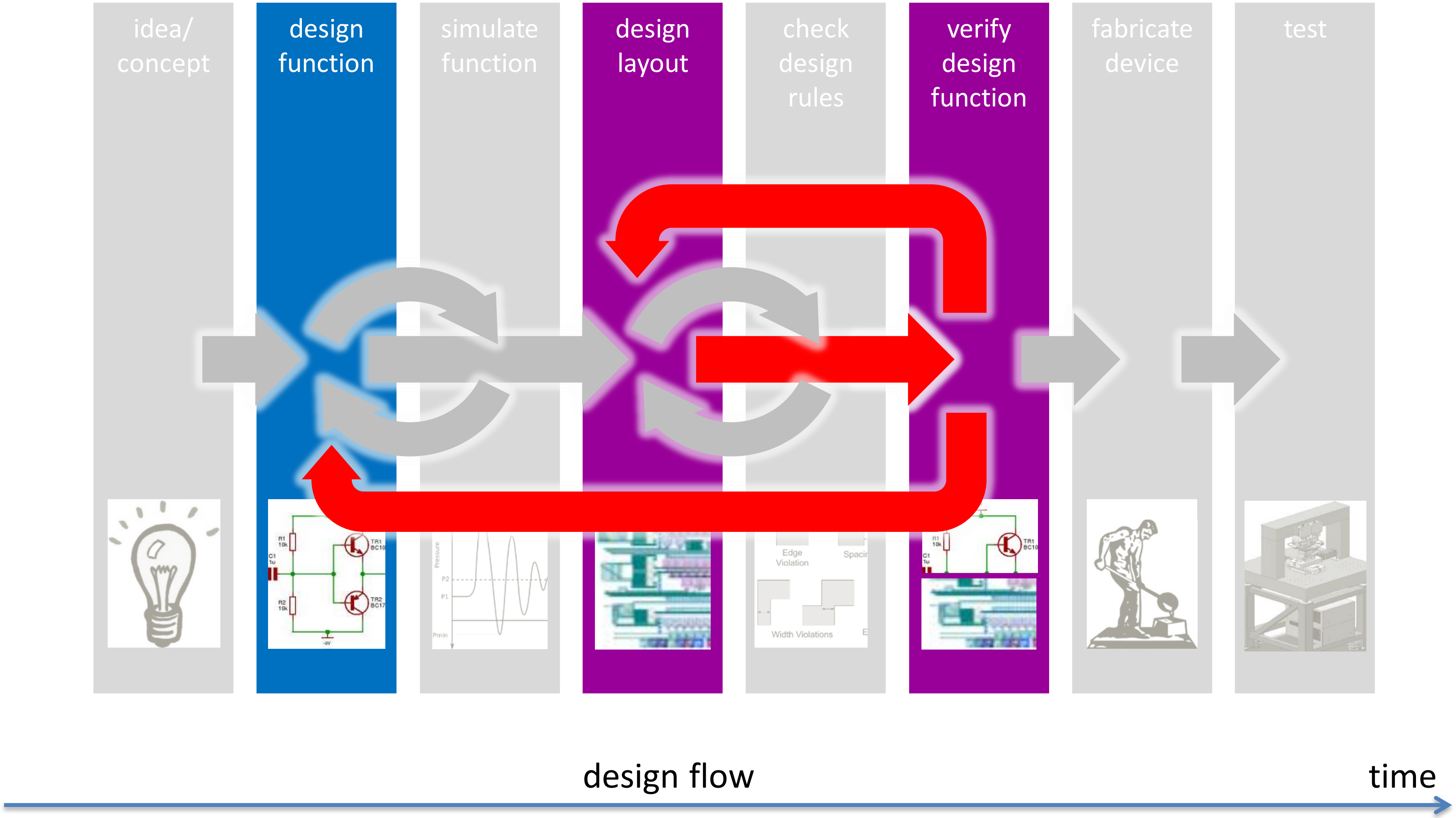
e.g. does the waveguide have the correct width (i.e. optical length)





GHENT UNIVERSITY · imec

FUNCTIONAL VERIFICATION

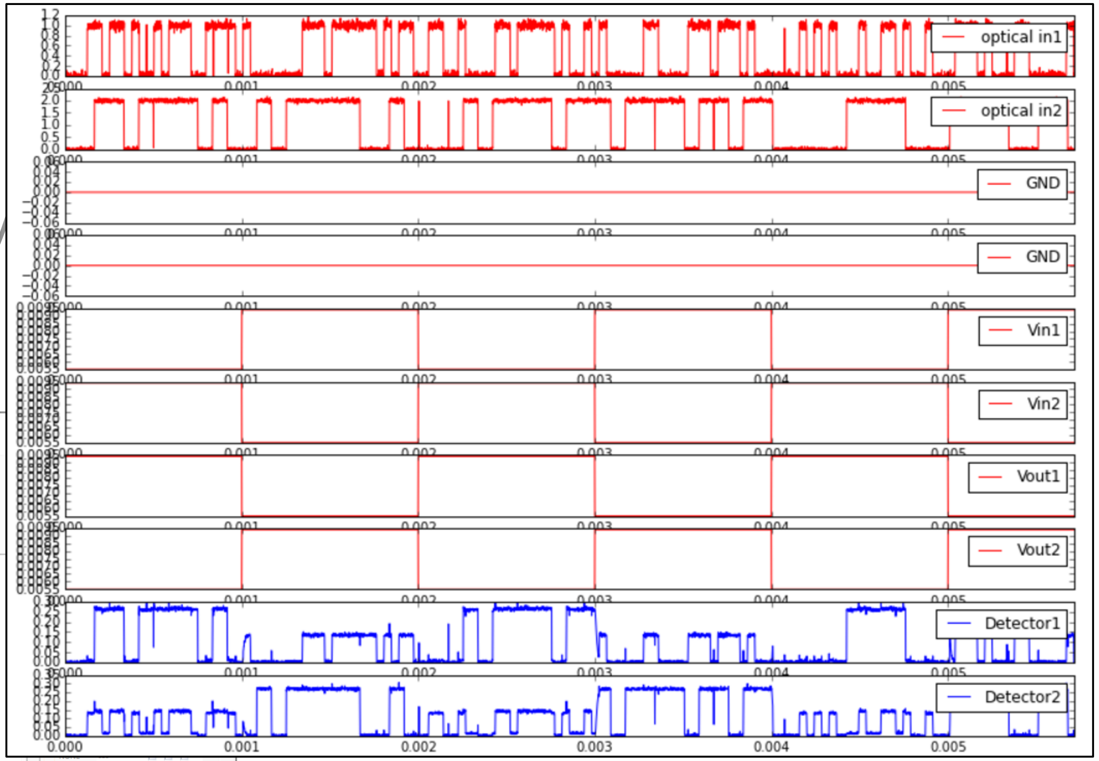


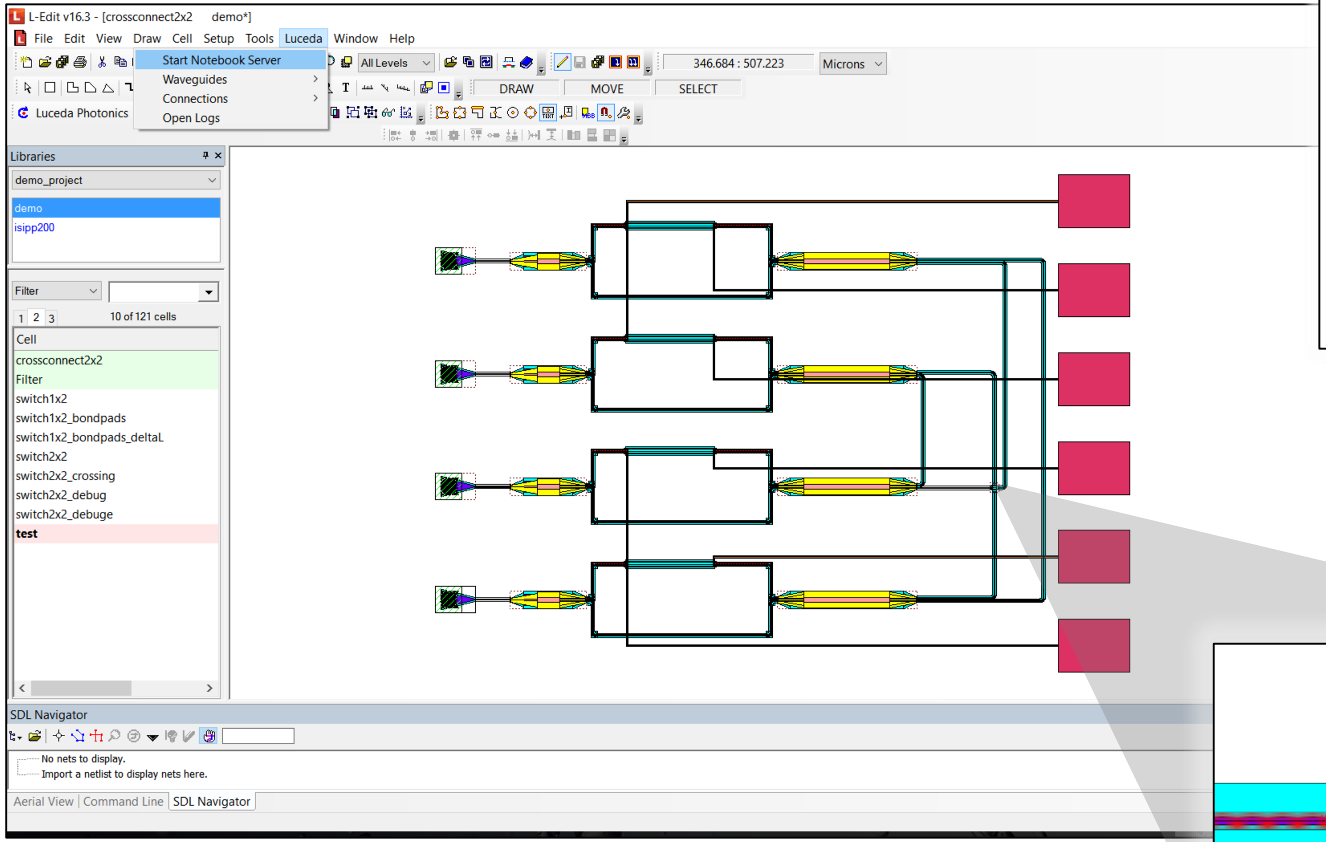
GHENT UNIVERSITY · imec

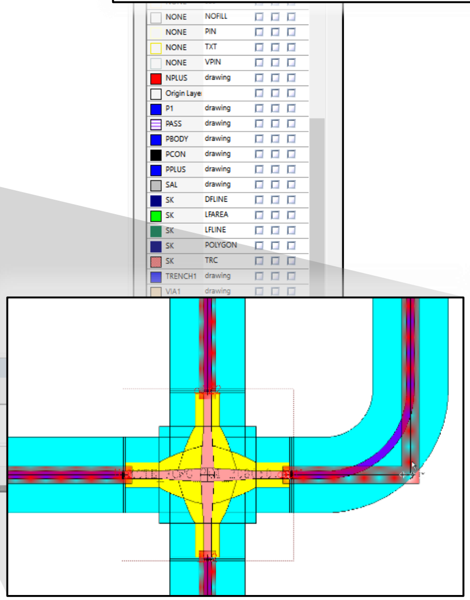
POST-LAYOUT SIMULATION

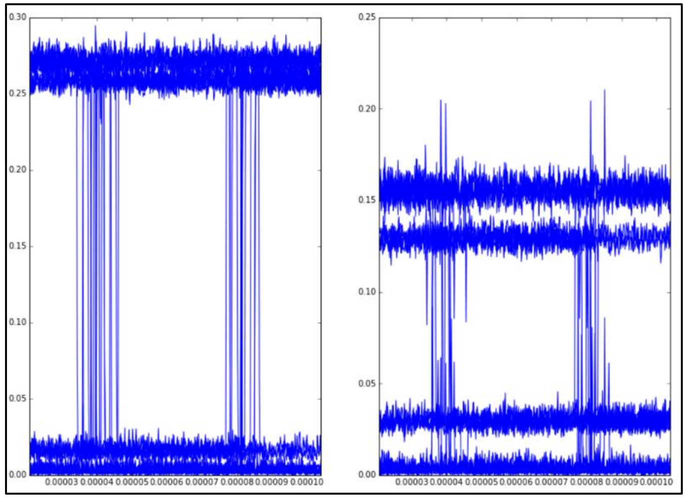
Resimulate the circuit based on the actual lay

Include lengths, crossings, reflections, …





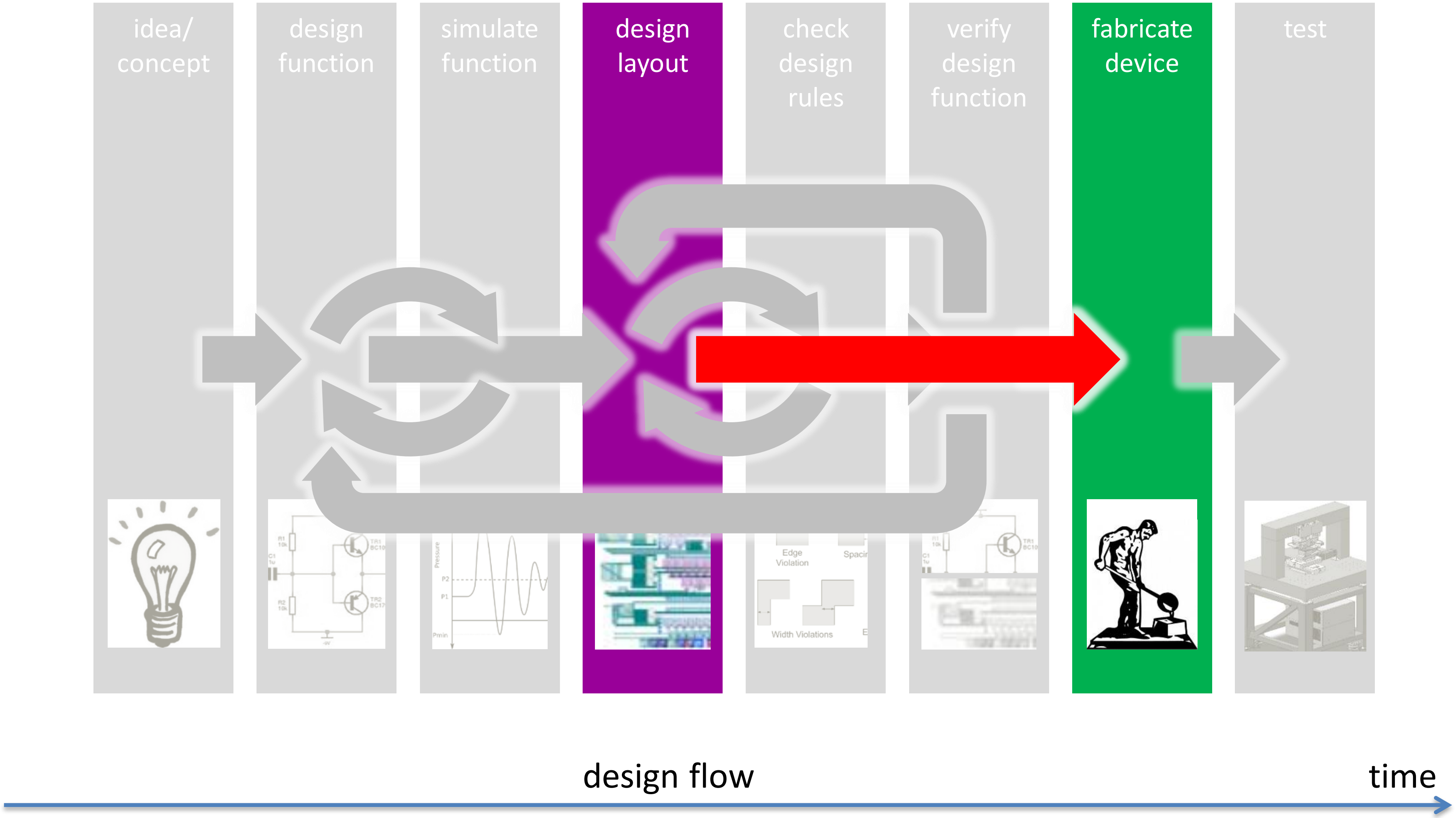




FABRICATION

“no plan survives contact with the enemy”

H. von Moltke (misquoted)



THE ACTUAL FABRICATION PROCESS

Layer depositions

Pattern definition (lithography)

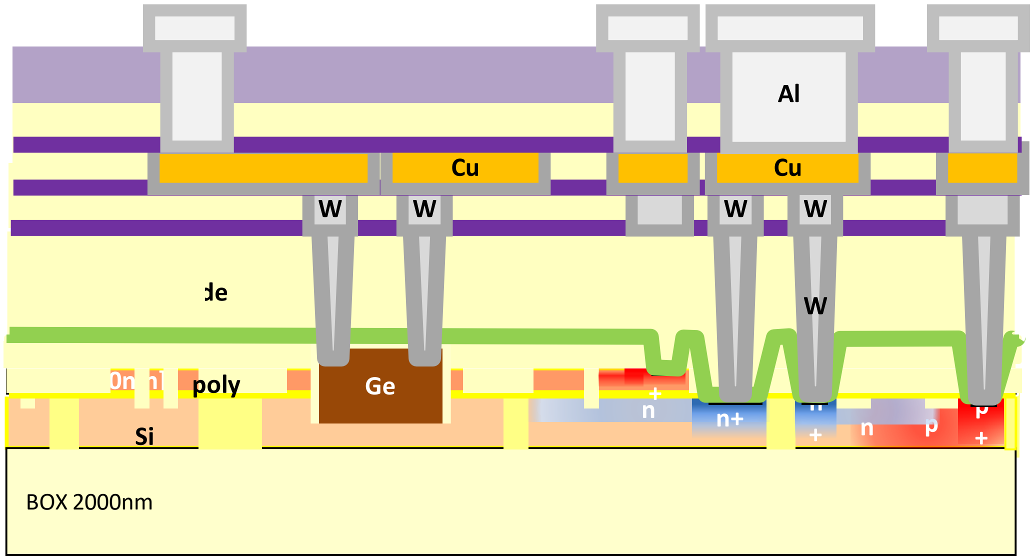
Pattern transfer (etch)

Planarization

Thermal treatment

Doping and implantation

…



example: IMEC silicon Photonics

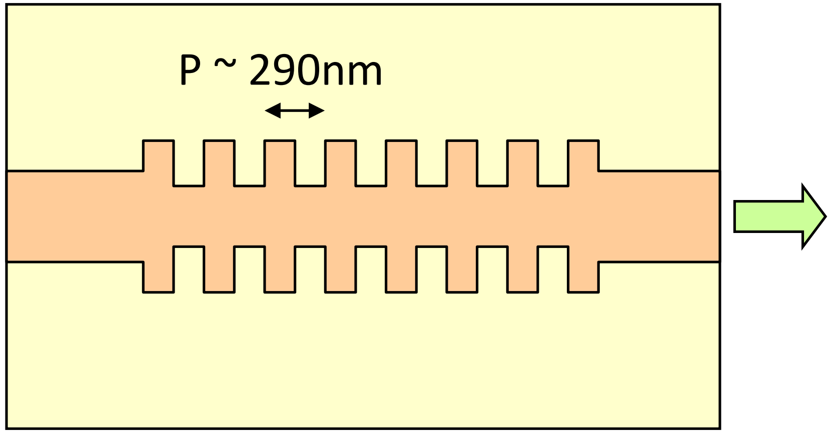
and each step with imperfections and variability

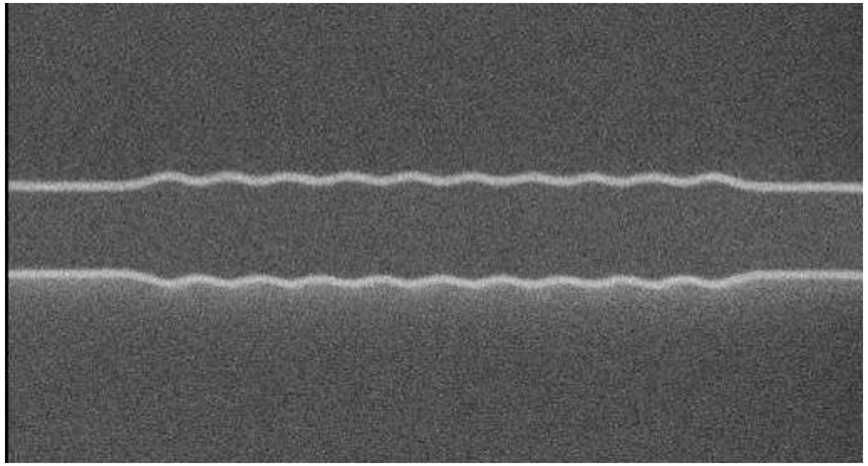
LITHOGRAPHY: NOT PERFECT

Spatial low-pass filter

- Minimum feature size
- Minimum pitch
- Pattern rounding

Example: Bragg grating





OPTICAL PROXIMITY CORRECTIONS (OPC)

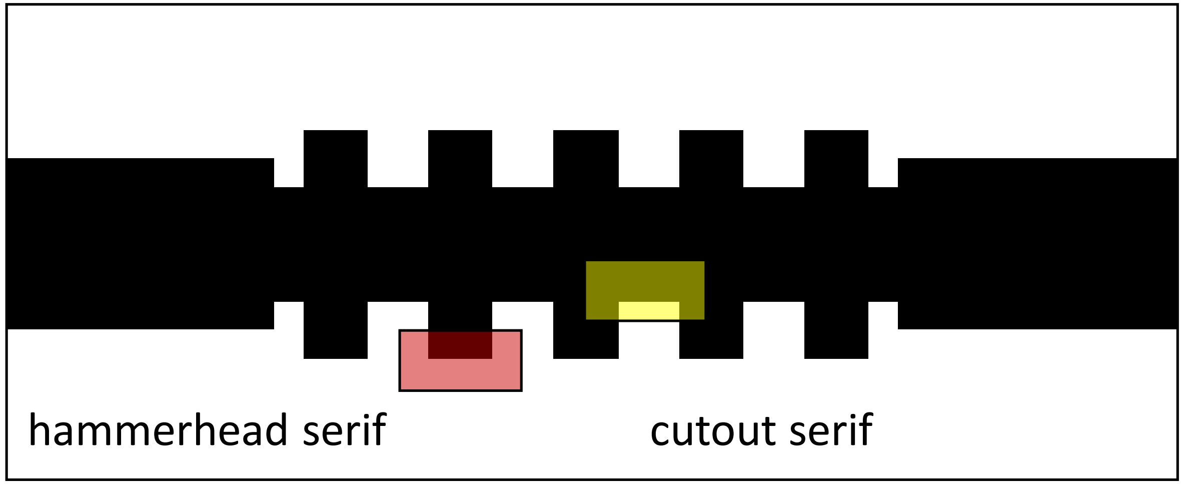
Overcome rounding: add OPC

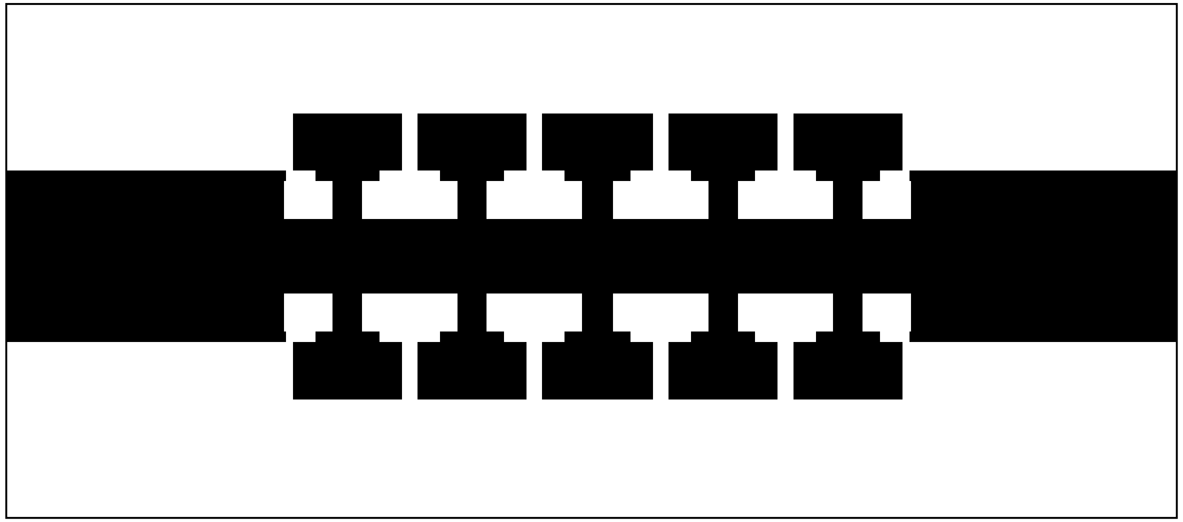
— serifs

— cutouts

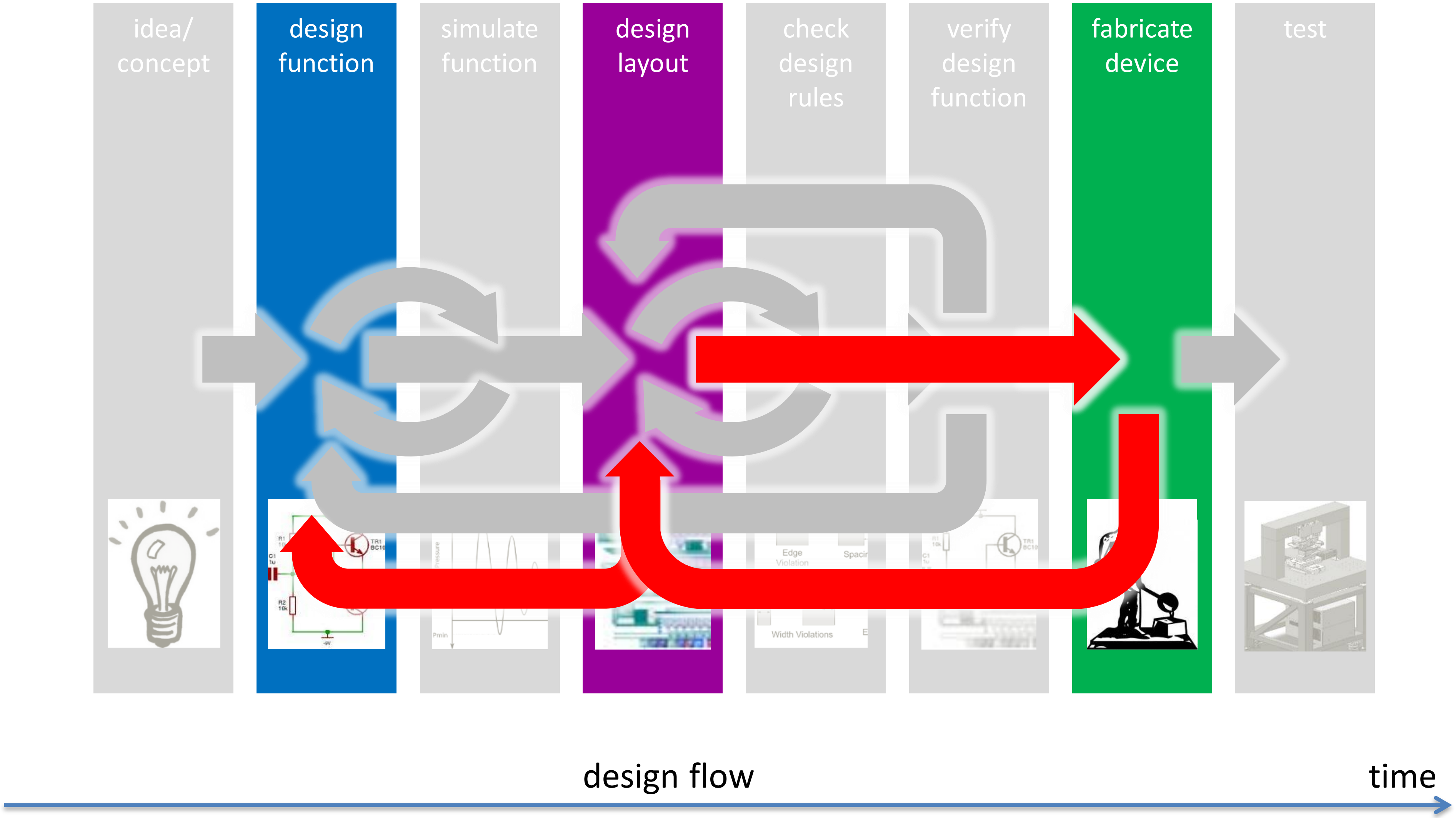
Makes mask more complex (and costly)

Not always possible without violating DR





FABRICATION: IN-LINE DATA



IN-LINE PROCESS DATA

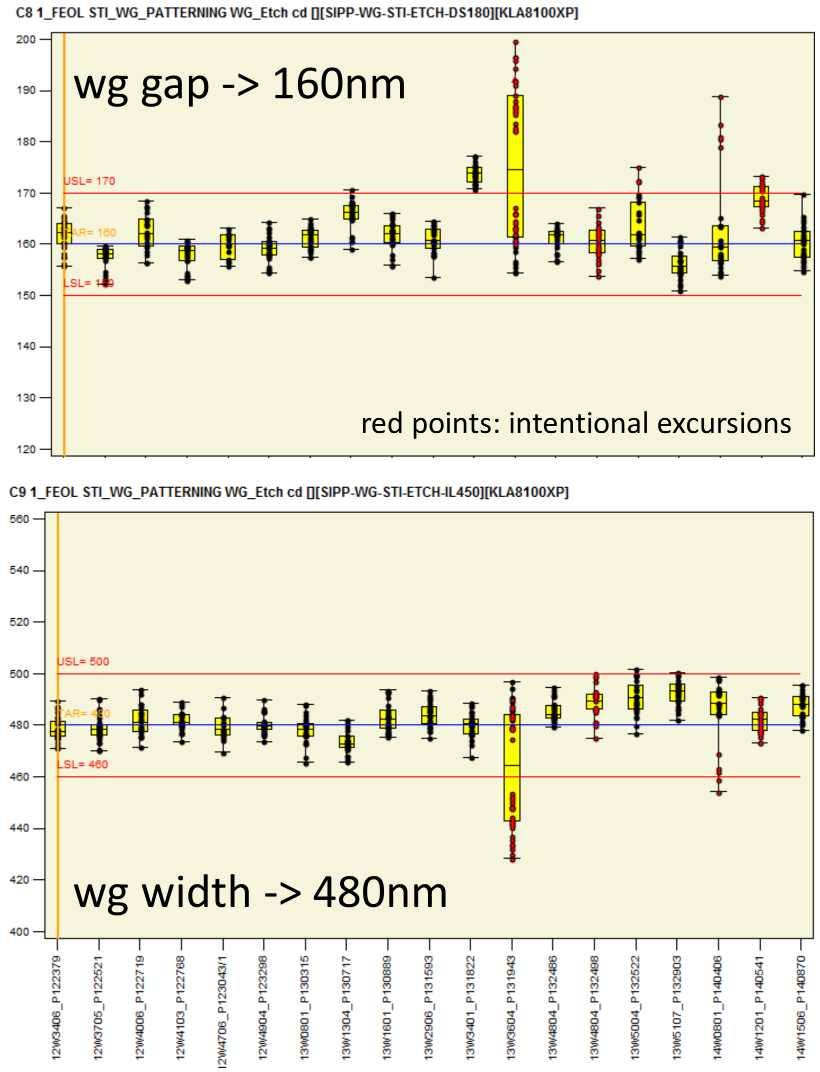
Collect data from wafers as they are being processed

- Line width
- Etch depth
- Layer thickness
- …

Feed in design process

- FRONT-END: Predict behavioural change
- BACK-END: Adjust layout

STATISTICS!



THERE ARE MANY SOURCES OF NON-UNIFORMITY

Reticle/Mask
- CD uniformity
- Flatness
- Transmission

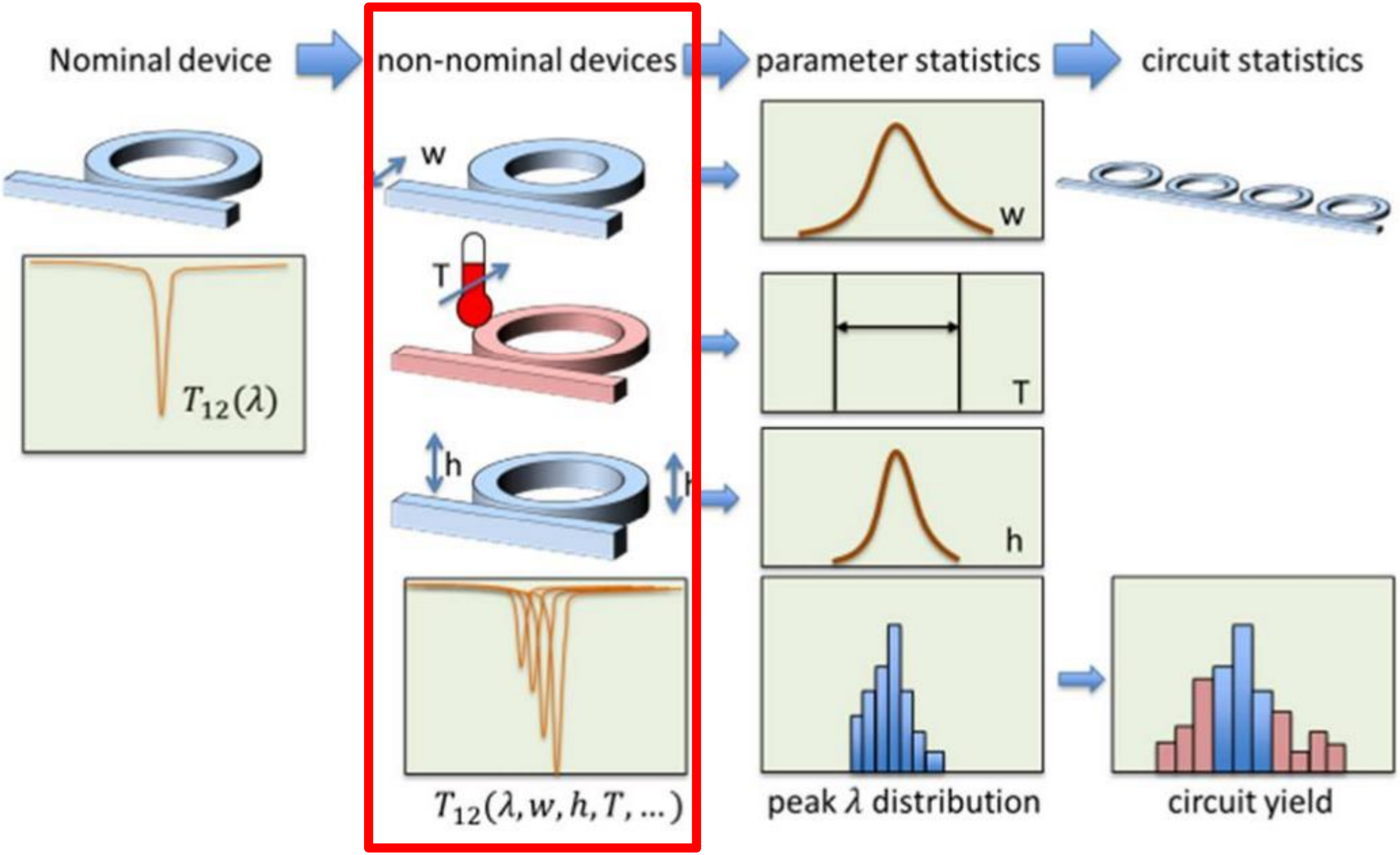
Litho tool
- Exposure dose
- Slit uniformity
- Chuck flatness
- Focus stability
- Scan direction
- Source spectrum

Resist process
- BARC uniformity
- Resist uniformity
- PEB °C Uniformity
- Developer
- Metrology

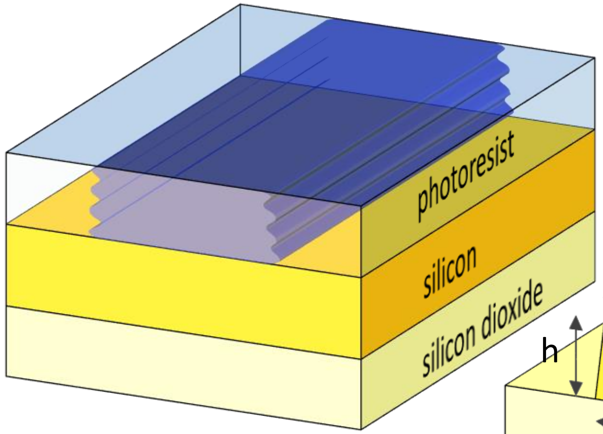
Wafer
- Wafer flatness
- Stack uniformity
- Topography

Etch process
- Plasma Chemistry
- Coil power stability
- Bias stability
- Resist coverage
- °C stability
- Metrology

VARIABILITY: PREDICTING CIRCUIT YIELD

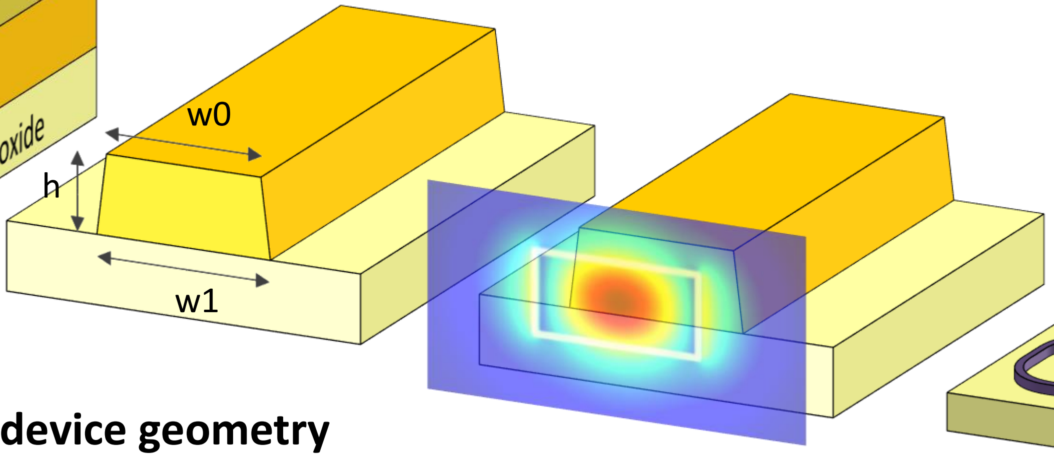


DESCRIBING VARIABILITY AT DIFFERENT LEVELS



process conditions

- exposure dose
- resist age
- plasma density
- slurry composition
- …

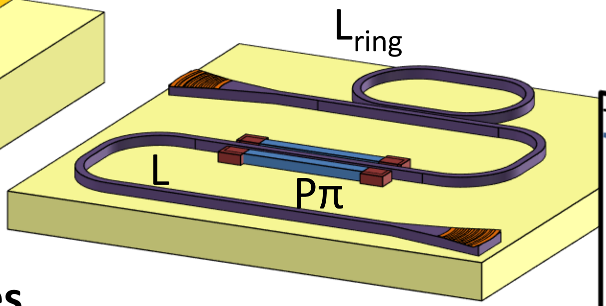


device geometry

- line width
- layer thickness
- sidewall angle
- doping profile
- …

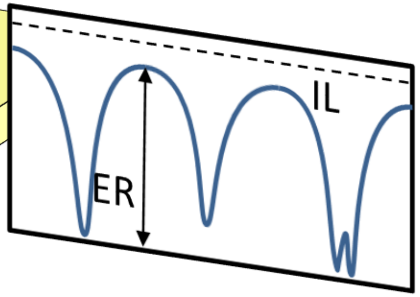
optical device properties

- effective index
- group index
- coupling coefficients
- center wavelength
- …



circuit properties

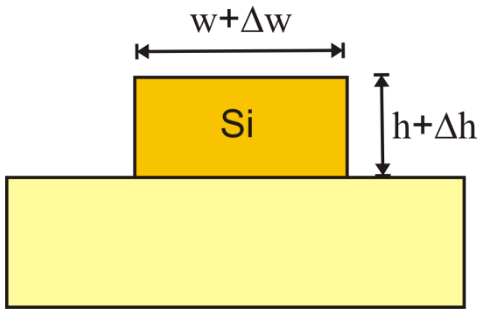
- optical delay
- path imbalance
- tuning curve
- …

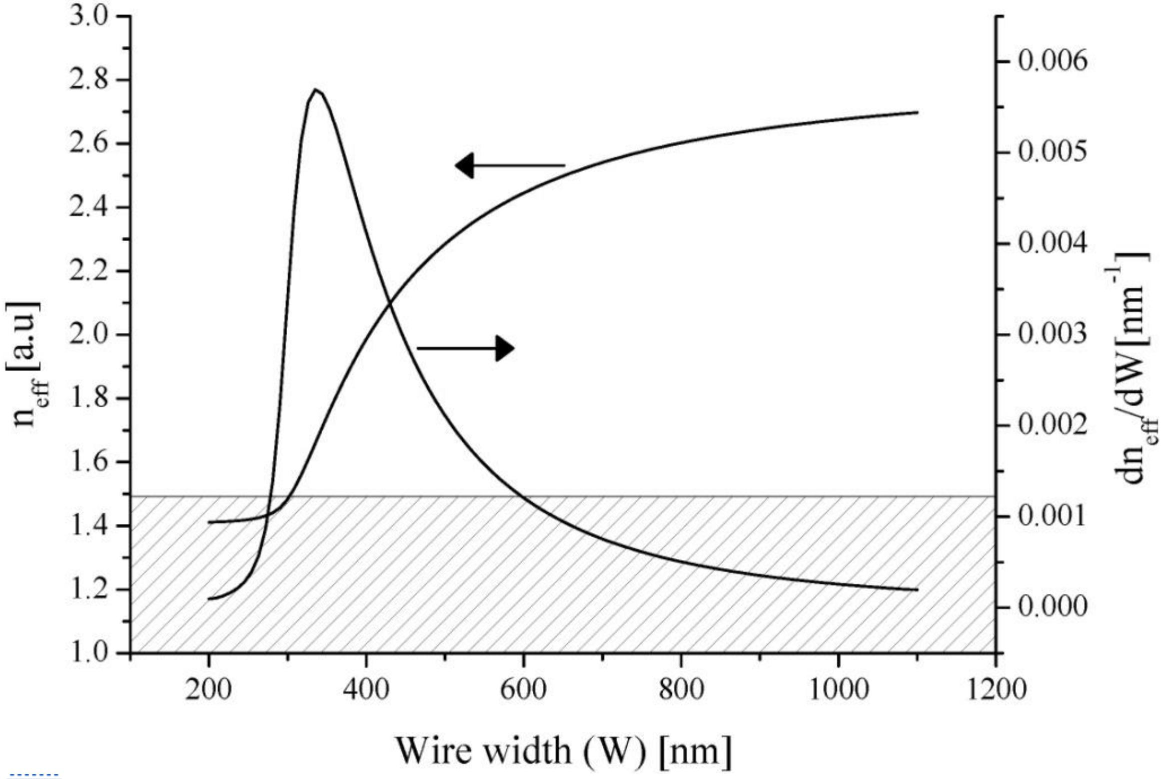


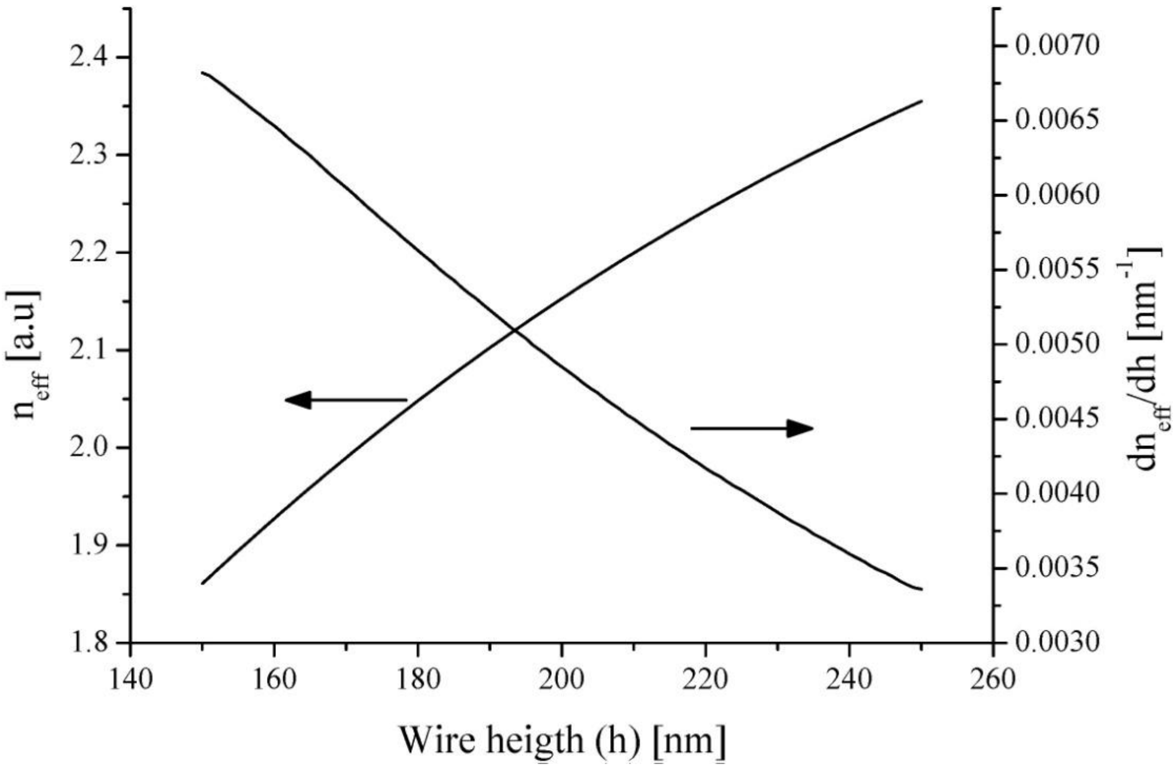
system performance

- insertion loss
- crosstalk
- noise figures
- power consumption
- …

DIMENSIONAL DEPENDENCE OF A WAVEGUIDE

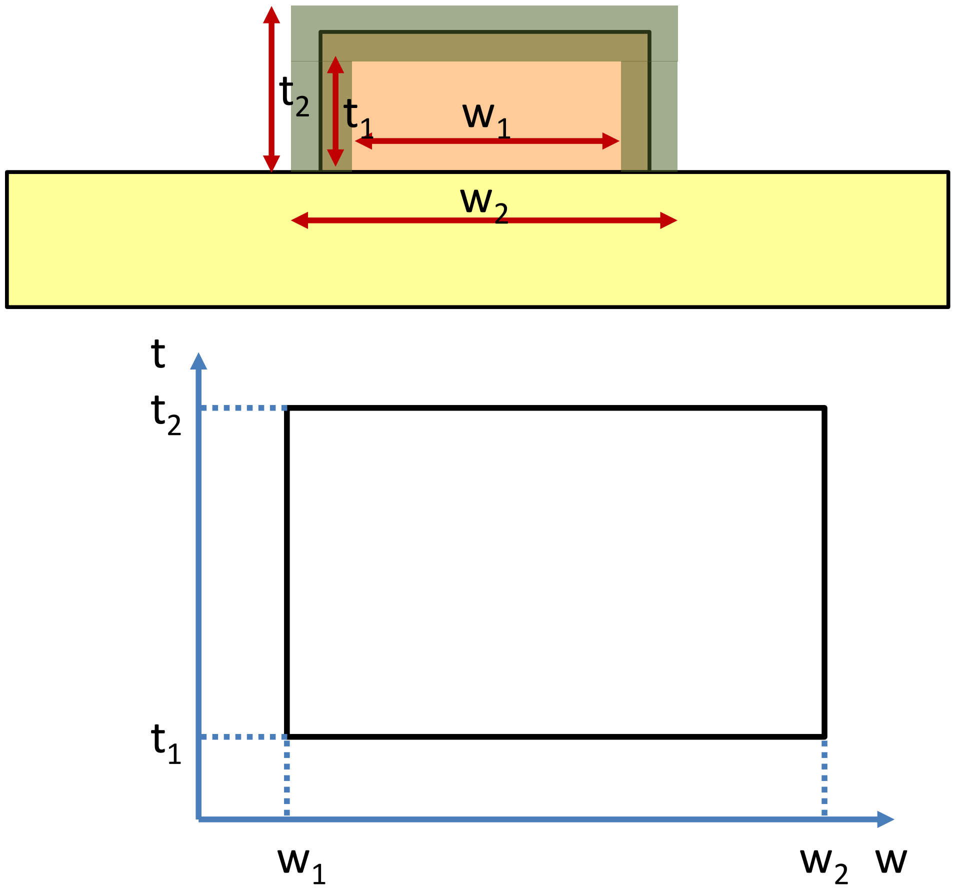






LEVELS OF VARIABILITY: CAREFUL WITH MAPPING

Geometry: width and thickness



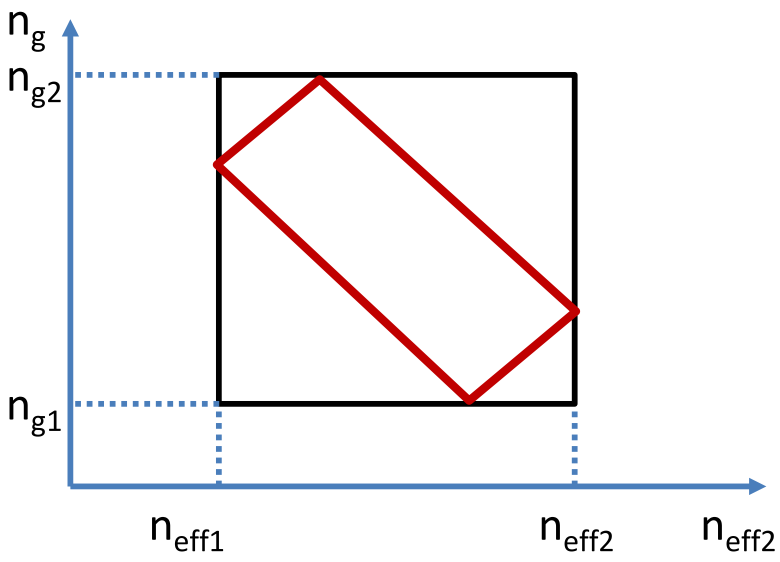
Model: n_{eff} and n_g

$w_1,t_1 \Rightarrow n_{eff1}$

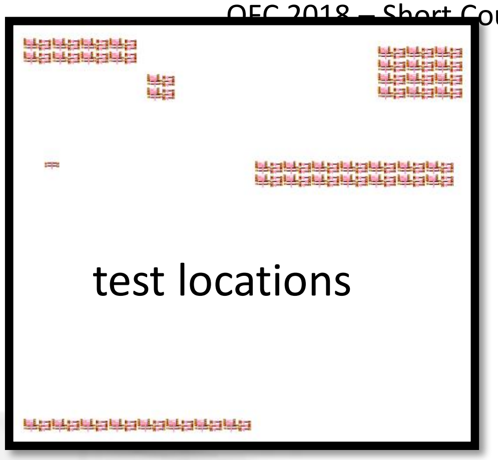
$w_2,t_2 \Rightarrow n_{eff2}$

$w_1,t_2 \Rightarrow n_{g2}$

$w_2,t_1 \Rightarrow n_{g1}$



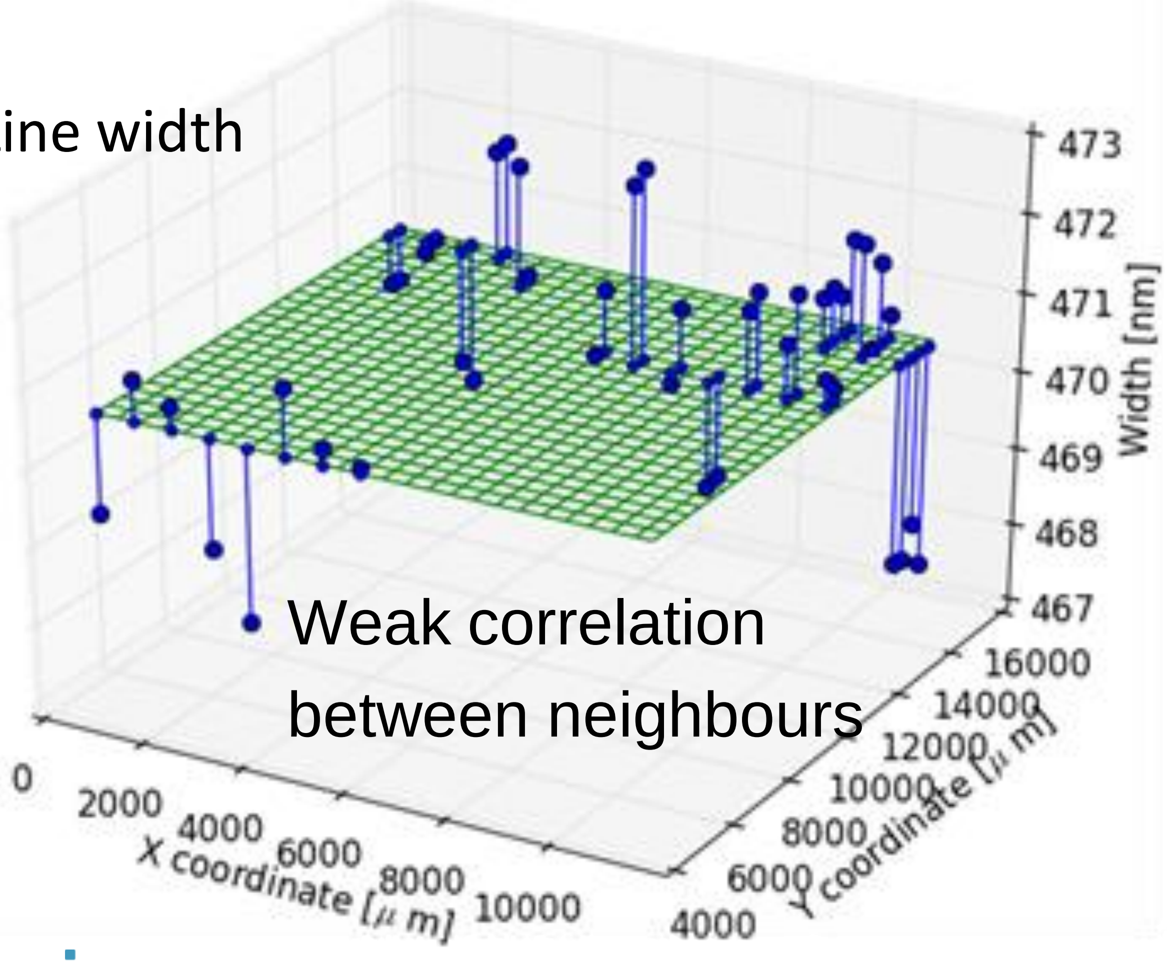
INTRA-DIE VARIABILITY



Variability has causes with different properties…

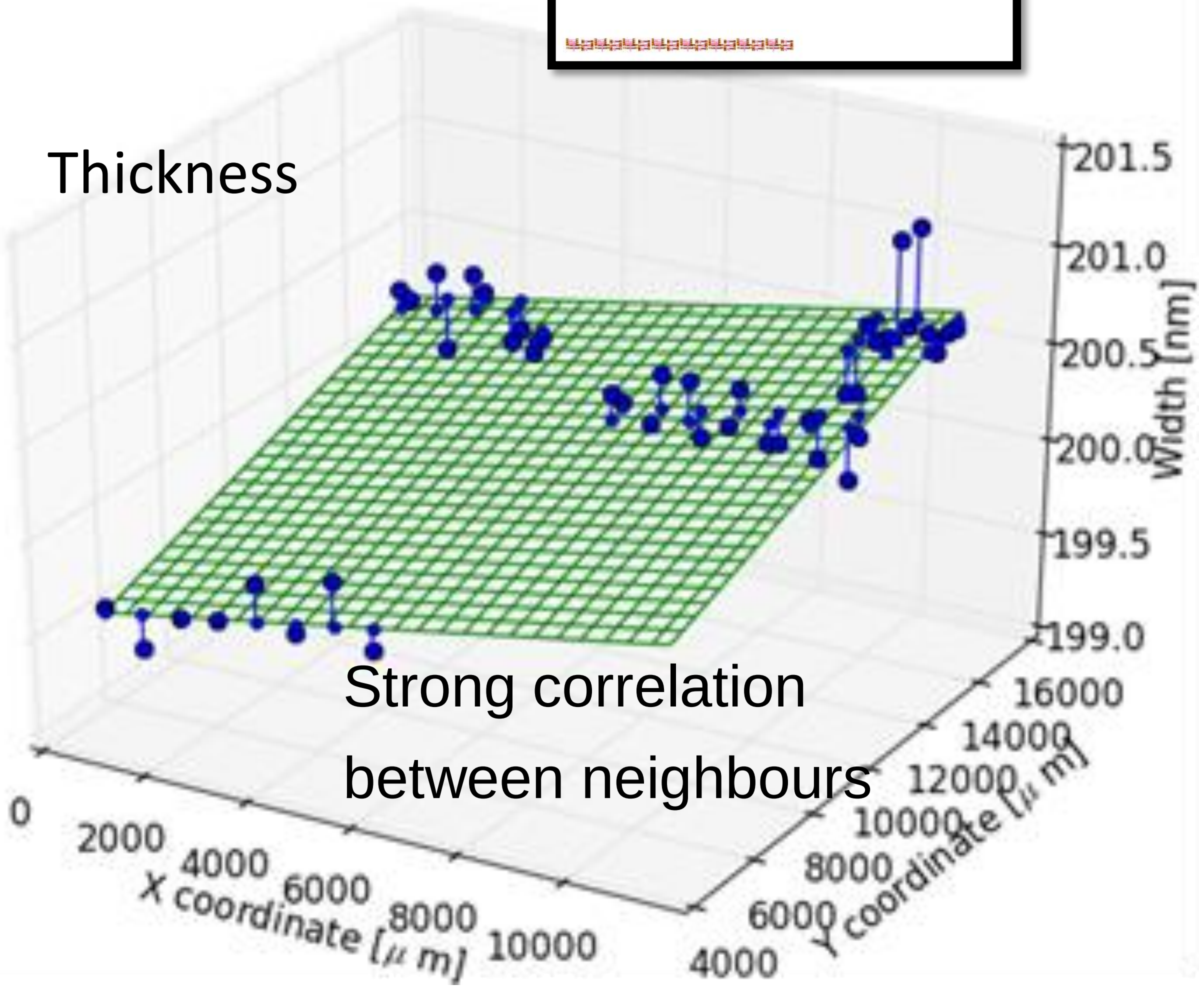
Optical extraction of linewidth and thickness

Line width



Weak correlation between neighbours

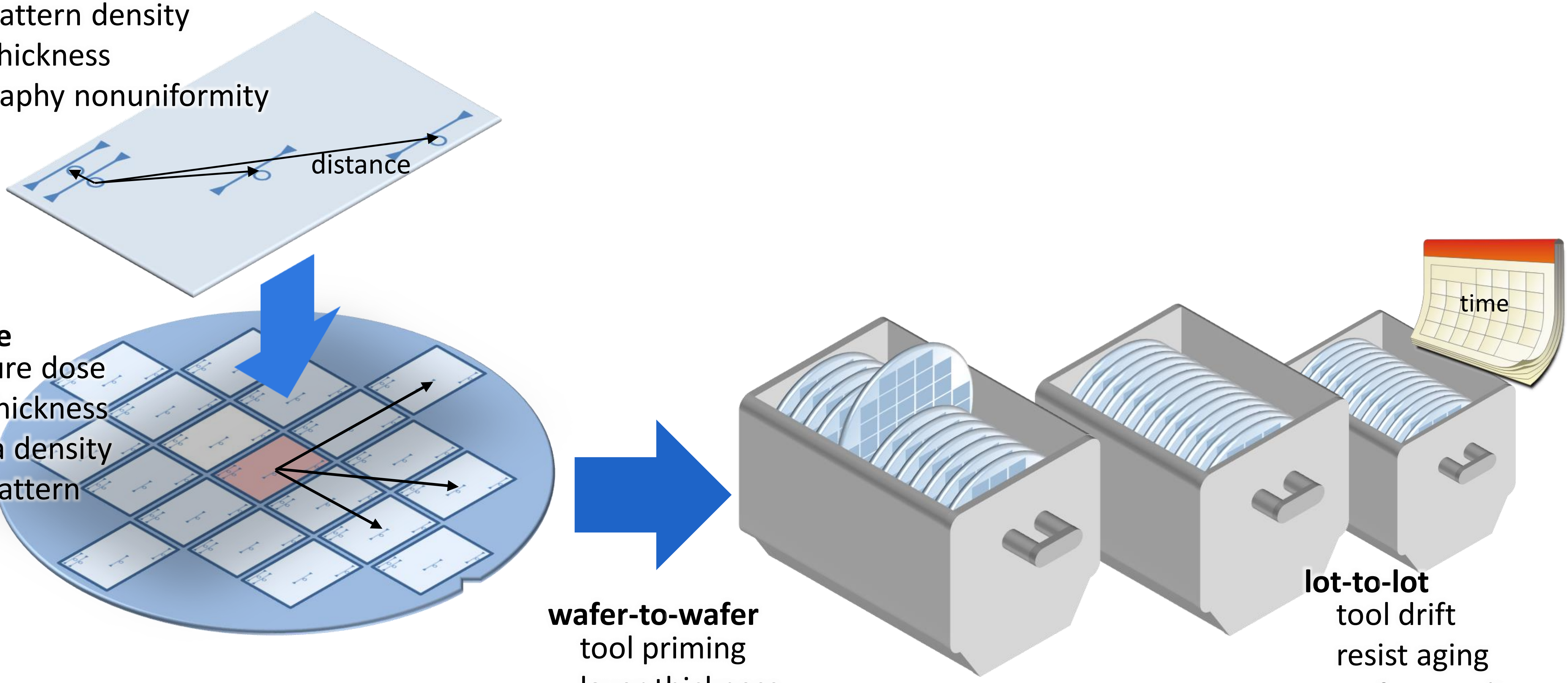
Thickness



Strong correlation between neighbours

VARIABILITY EFFECTS WORK ON DIFFERENT SCALES

intra-die
local pattern density
layer thickness
lithography nonuniformity



die-to-die
exposure dose
layer thickness
plasma density
CMP pattern

wafer-to-wafer
tool priming
layer thickness

lot-to-lot
tool drift
resist aging
wafer supplier

GHENT UNIVERSITY imec

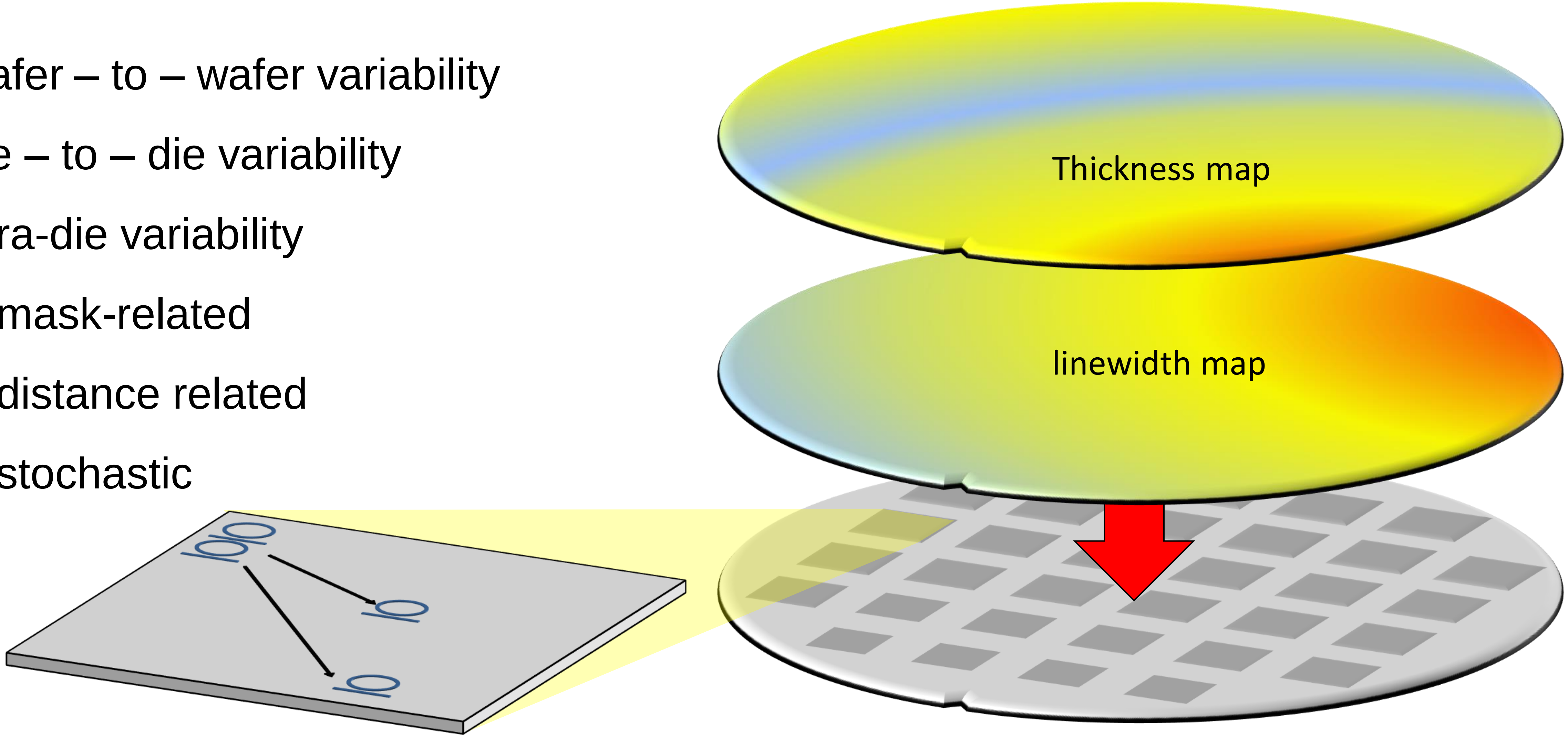
VARIABILITY ≠ VARIABILITY

Wafer – to – wafer variability

Die – to – die variability

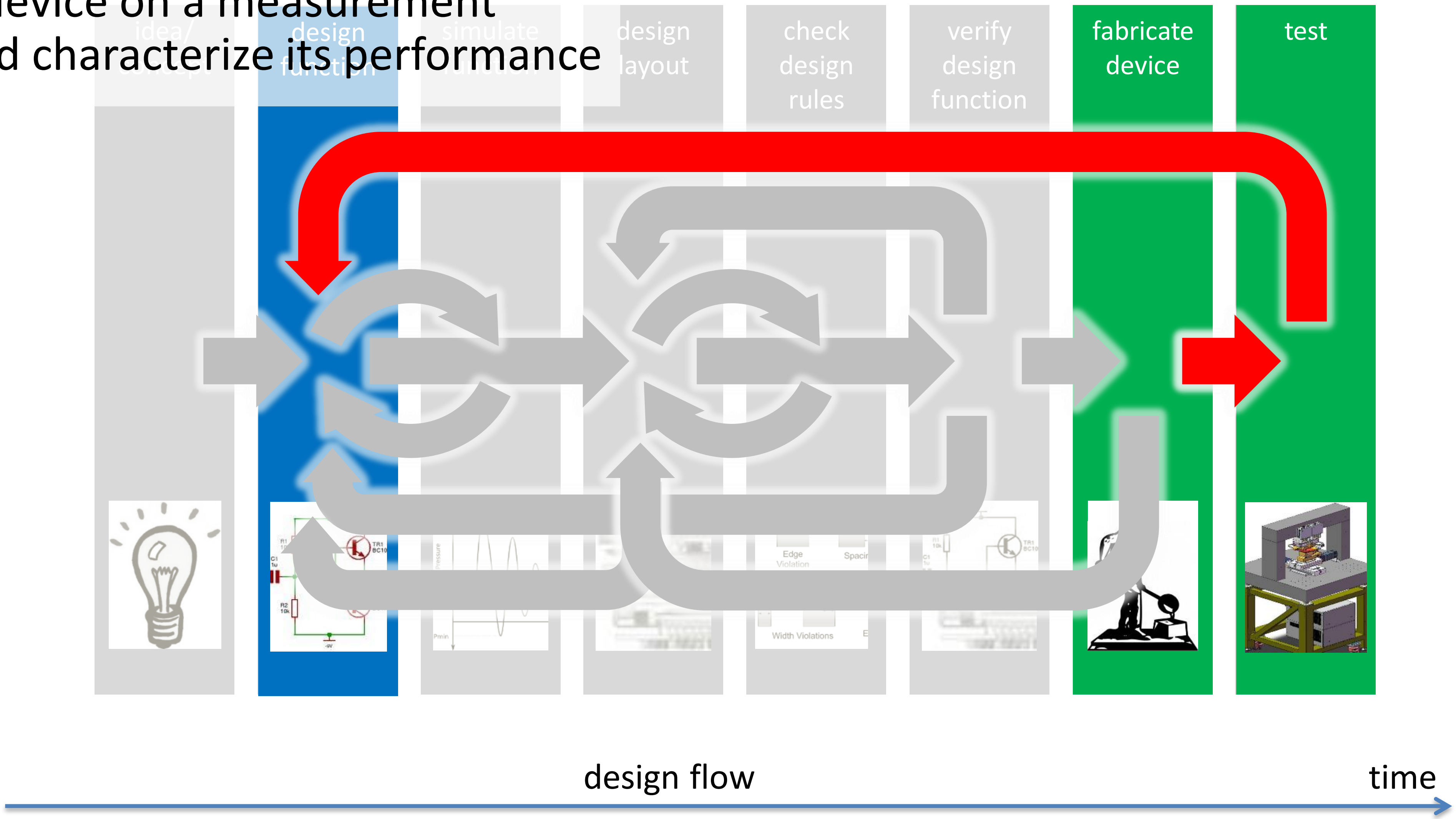
Intra-die variability

- mask-related
- distance related
- stochastic



TESTING

Put the device on a measurement setup and characterize its performance



GHENT UNIVERSITY
imec

How to test?

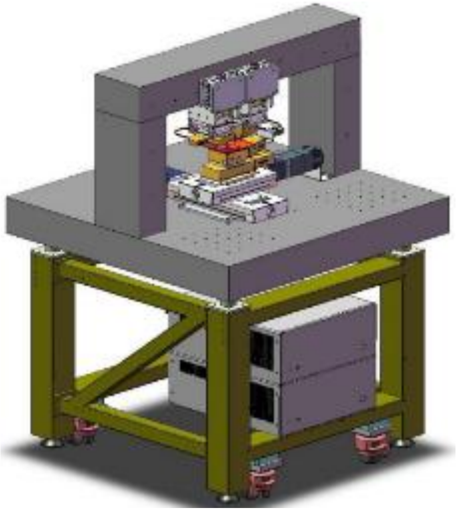
Electrical, optical, or both?

Wafer-scale testing -> grating couplers

Testing after packaging?

Need statistics?

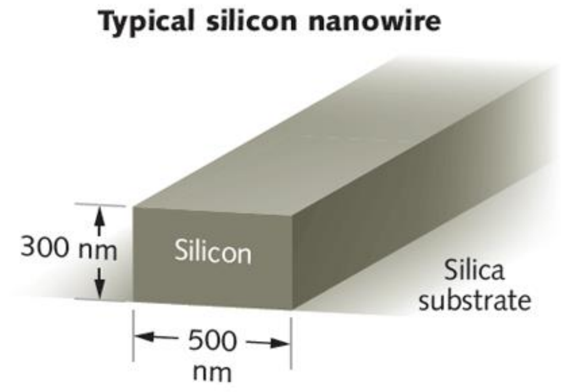
depends



CHALLENGE: DEFINING GOOD TESTS

You need to think about tests during the design stage

- Which structures are representative?
- How can I isolate them?
- What parameters do I want to measure?
- How will I analyse/fit the data?



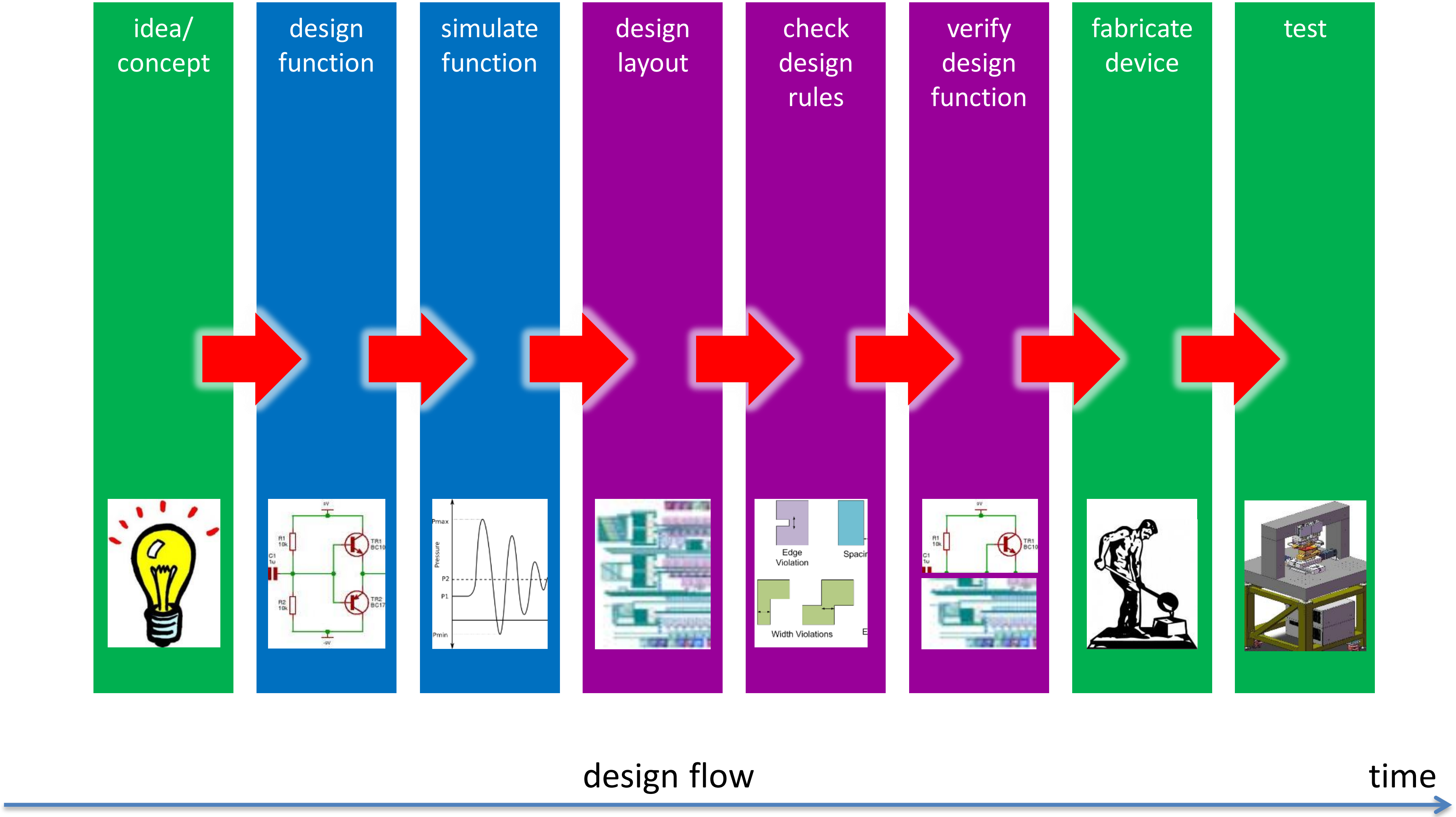
Example: waveguide model
- $n_{eff}(\lambda)$ -> polynomial?
- loss(λ) -> polynomial?
- nonlinearities?

How to measure n_{eff}?

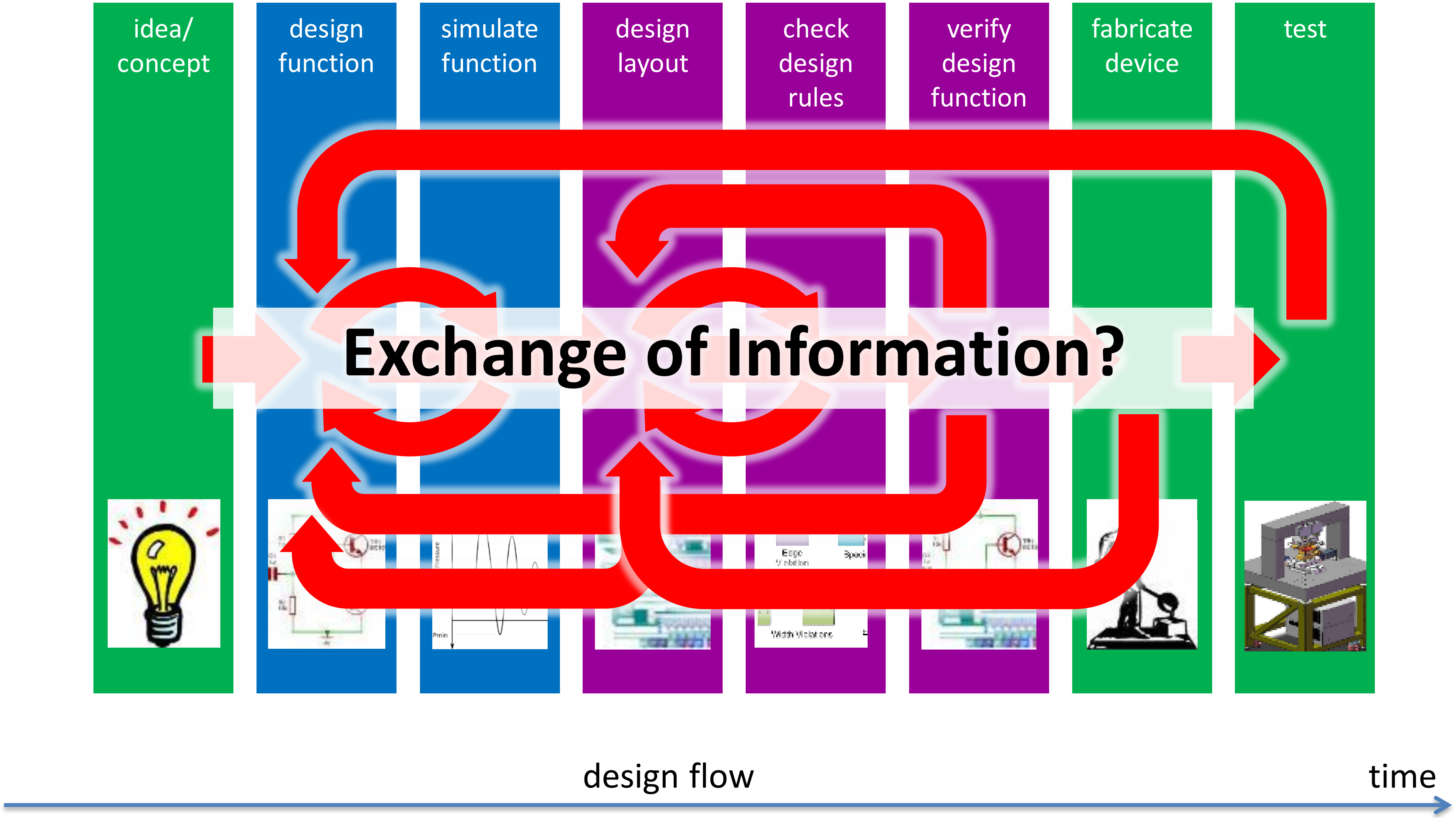
Parameters for your component models!

- What makes a good model?

OUR SIMPLE DESIGN FLOW



OUR SIMPLE DESIGN FLOW



EXCHANGE OF INFORMATION

Files

- Layout: GDSII and OASIS
- Netlist/Schematic: Spice, EDIF
- Models: Spice, VerilogA, C++, Python
- PCell code: Skill, Python , Tcl
- Data: Touchstone, XML

Databases

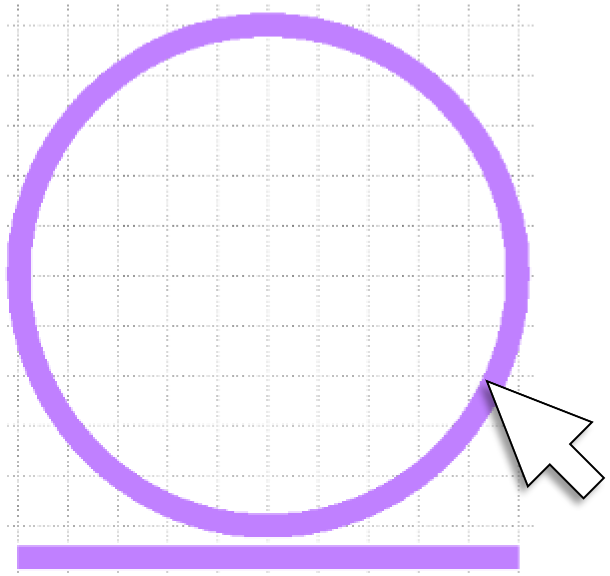
- proprietary
- EDA standard: OpenAccess

DESIGNING IN CODE VERSUS GUI

Designing in Code

```python
from ipkiss3 import all as i3

class RingResonator(i3.PCell):

    class Layout(i3.LayoutView):

        ring_radius = i3.PositiveNumberProperty(default=20.0)
        wg_width = i3.PositiveNumberProperty(default=0.45)
        coupler_gap = i3.PositiveNumberProperty(default=0.3)

        def _generate_elements(self, elems):
            r = self.ring_radius
            g = self.coupler_gap
            w = self.wg_width

            elems += i3.CirclePath(layer=i3.Layer(2),
                                   radius=r,
                                   line_width=w)
            elems += i3.Line(layer=Layer(2),
                             begin_coord=(-r, -r-w-g),
                             end_coord=(+r, -r-w-g),
                             line_width=w)
            return elems
```

Designing in GUI



DESIGNING IN CODE VERSUS GUI

Designing in Code

Pro:

- Easy to reuse
- Easy to upgrade design
- Easy to share and version
- Easy to parametrize
- Easy to document and make examples
- Everything is numerically correct
- Automate repetitive work

Con:

- Harder to learn
- No immediate visual feedback

Designing in GUI

Pro:

- Intuitive quick start
- Visual feedback
- WYSIWYG
- Quick point and click

Con:

- Difficult to make complex things
- No calculations
- A lot of manual work
- Easy make small (invisible) mistakes

DESIGNING IN CODE VERSUS GUI

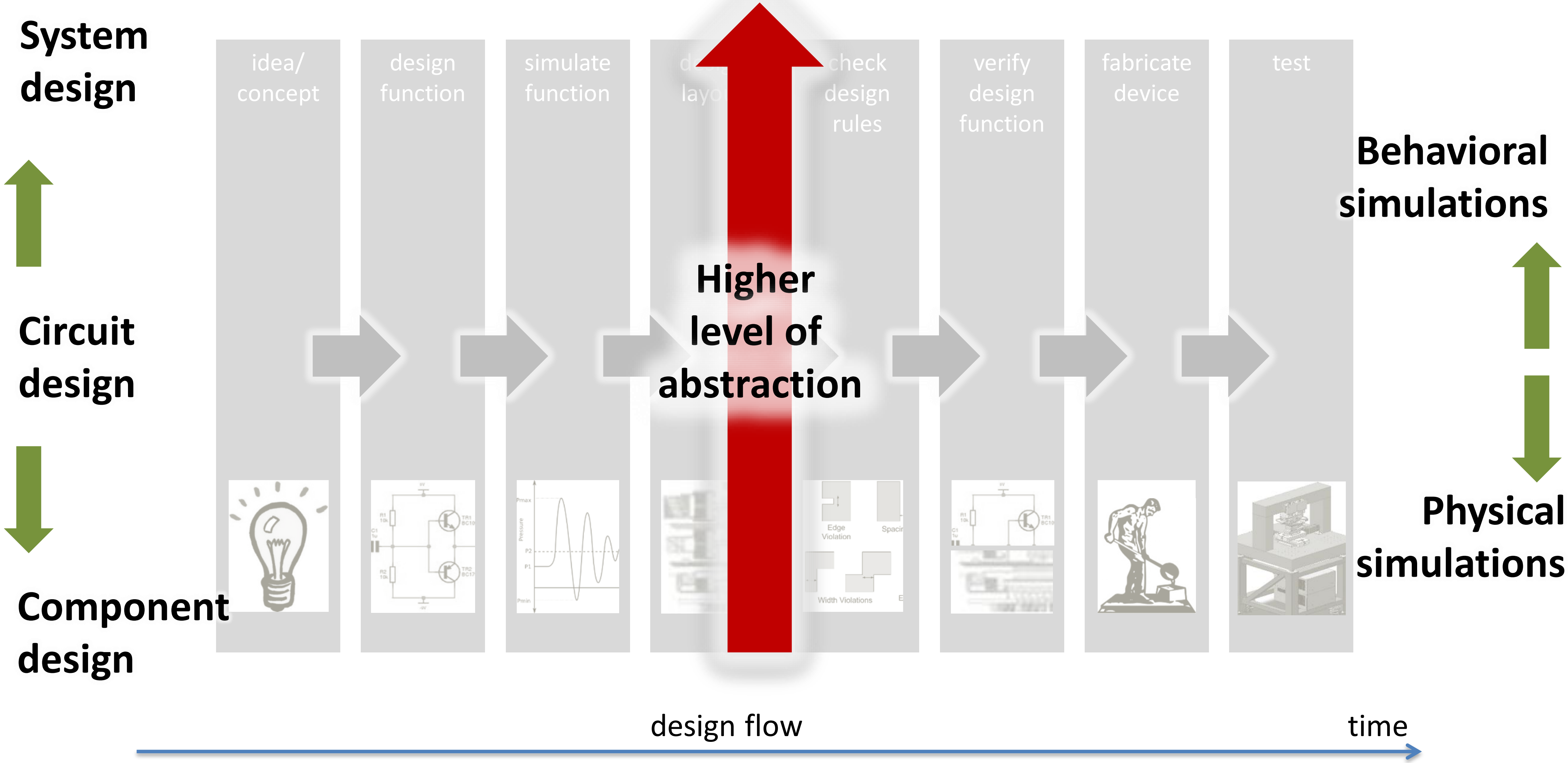
Designing in Code

- parameter sweeps
- calculated geometries
- circuit models
- automatic placement and routing

Designing in GUI

- schematic connectivity
- layout positioning (floorplanning)
- fixing the last DRC errors
- quick manual routing

DESIGN ABSTRACTIONS

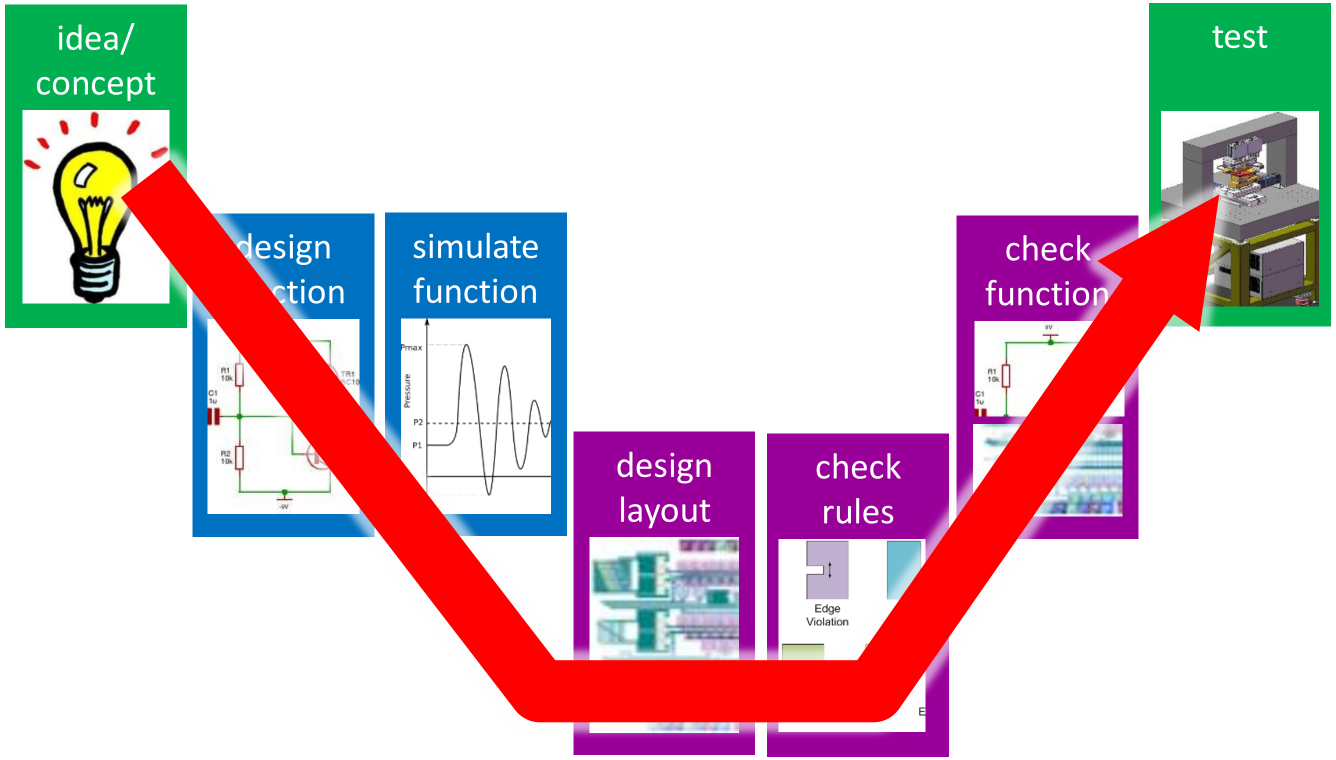


ABSTRACTIONS IN A CIRCUIT DESIGN FLOW

System design

Circuit design

Component design



Behavioral simulations

Physical simulations

design flow

time

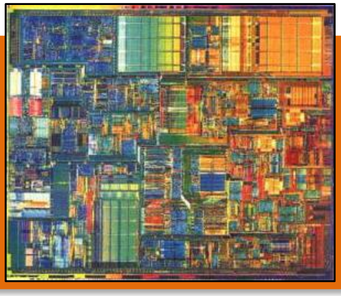
ABSTRACTIONS IN A CIRCUIT DESIGN FLOW

System design

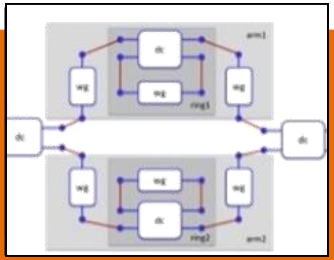
Circuit design

Component design

System

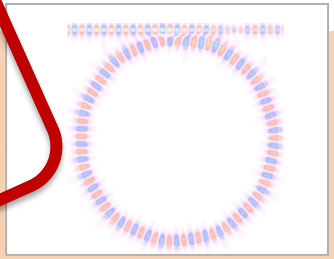


Circuit



Components

HANDLED BY THE FAB



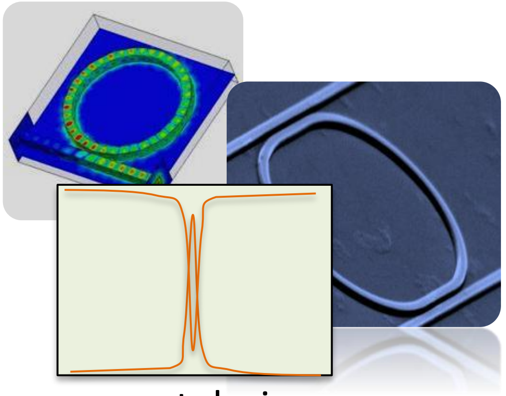
Behavioral simulations

Physical simulations

design flow

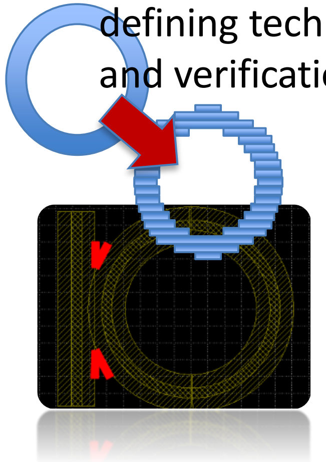
time

PDK: INTERFACE FROM FAB TO DESIGNER



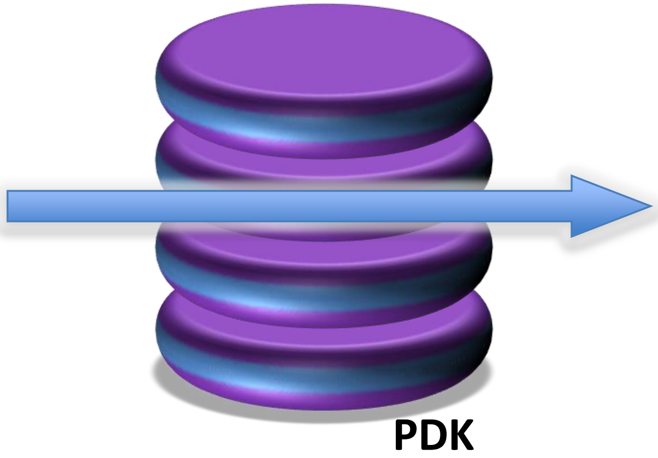
component design
simulation, measurement

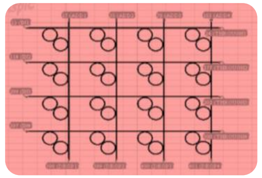
FAB



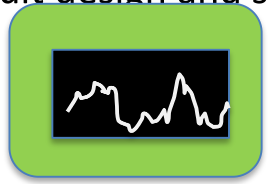
defining technology
and verification rules

component libraries
documentation
support scripts

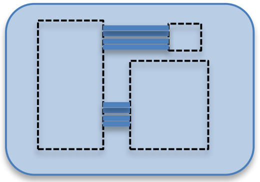




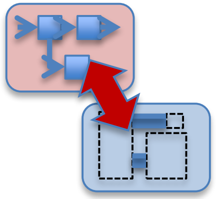
circuit design and simulation



DESIGNER



layout generation and verification



PDK for photonics

GHENT UNIVERSITY imec

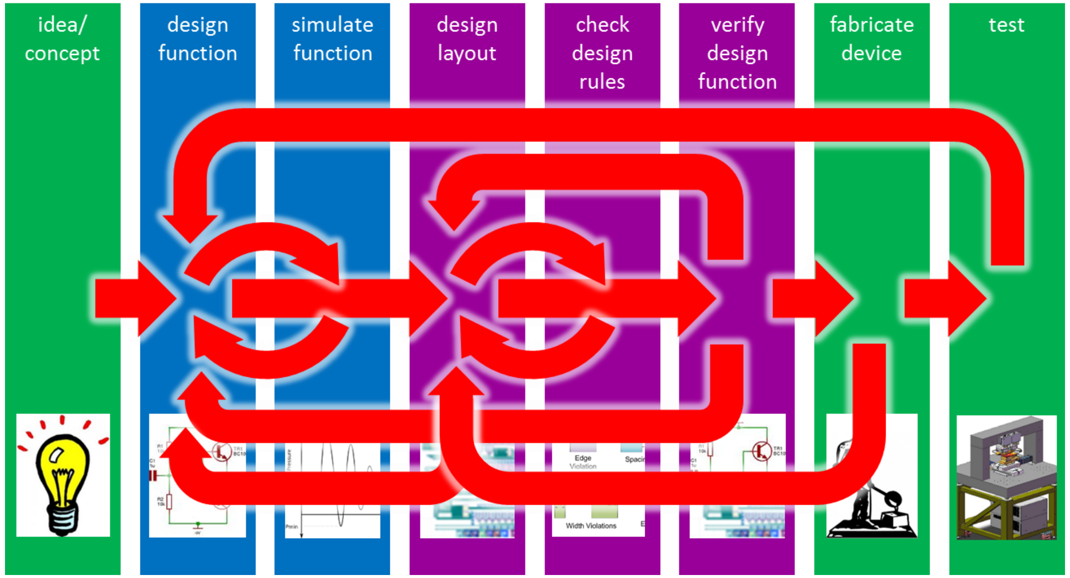
SUMMARY

(Silicon) Photonics is growing towards a circuit platform

- Technology supports larger circuits
- A circuit-oriented design flow is emerging (similar to electronics)
- Fabs are building PDKs

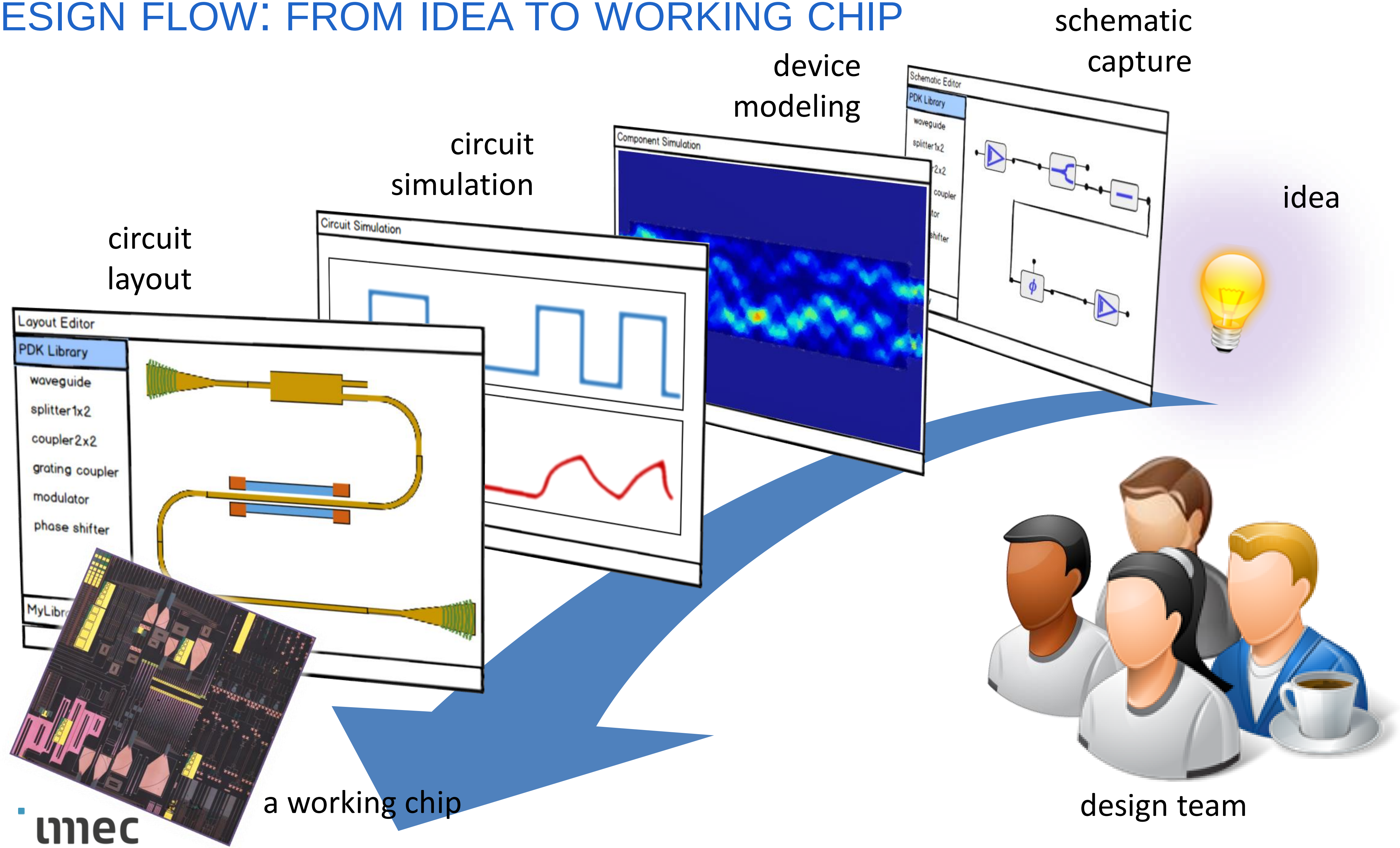
Challenges

- Schematic-driven Layout for photonics
- Variability: fabrication, performance, models
- Verification: DRC and LVS
- Design for manufacturability
- Photonic-electronic-software stacks



GHENT UNIVERSITY imec

DESIGN FLOW: FROM IDEA TO WORKING CHIP



GHENT UNIVERSITY imec

PRACTICAL SETUP

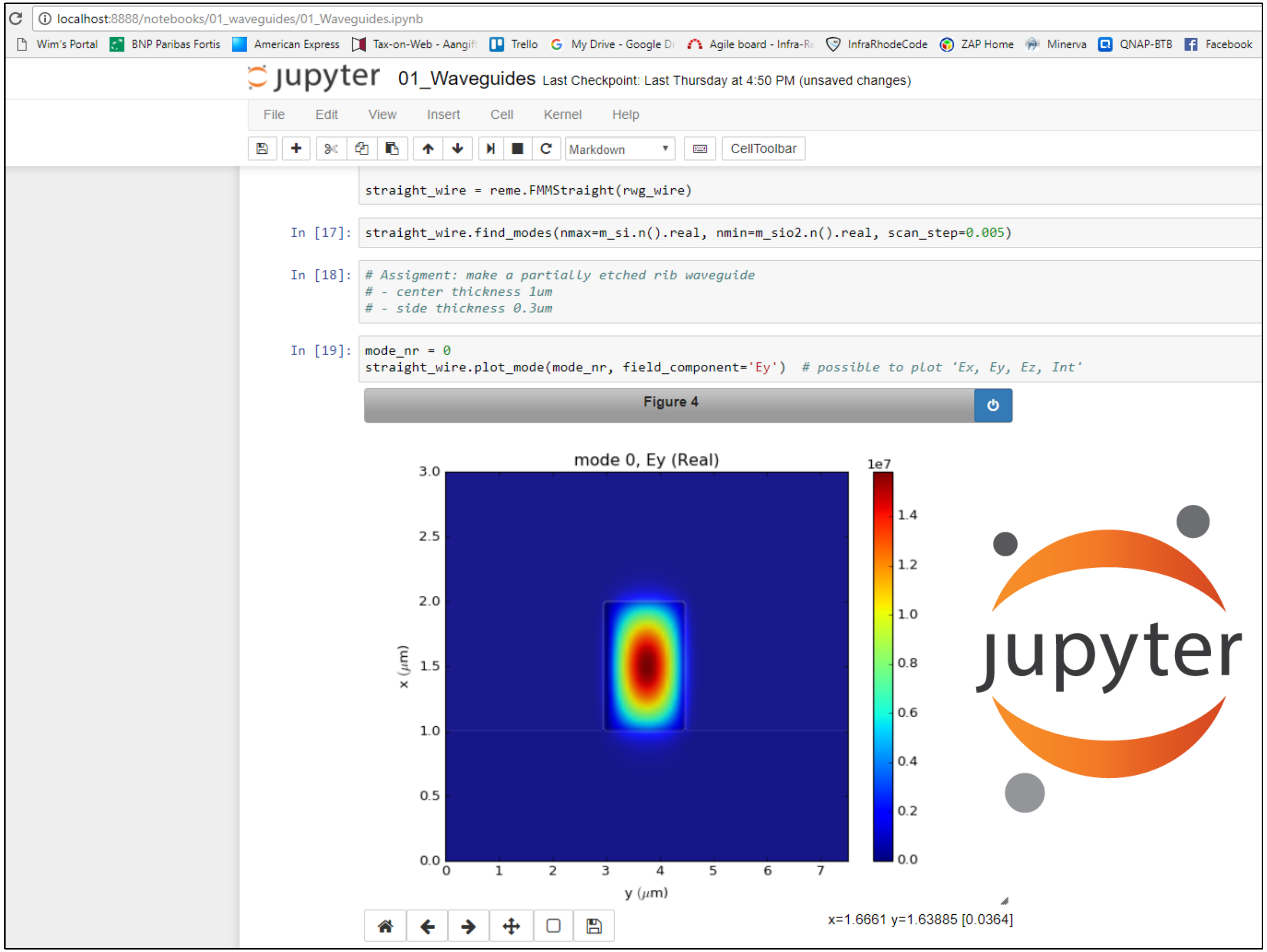
JUPYTER NOTEBOOKS

interactive notebook

- text, figures
- formulas
- python code

simulation and design

- built-in IPKISS



THE IPKISS DESIGN FRAMEWORK

Design framework for Photonic Integrated Circuits

- Parametric design
- Focus on reuse and automation

History

- Developed at Ghent University – imec in 2000-2014
- Spin-off into Luceda Photonics in 2014
- Currently hundreds of users worldwide

THE IPKISS DESIGN FRAMEWORK

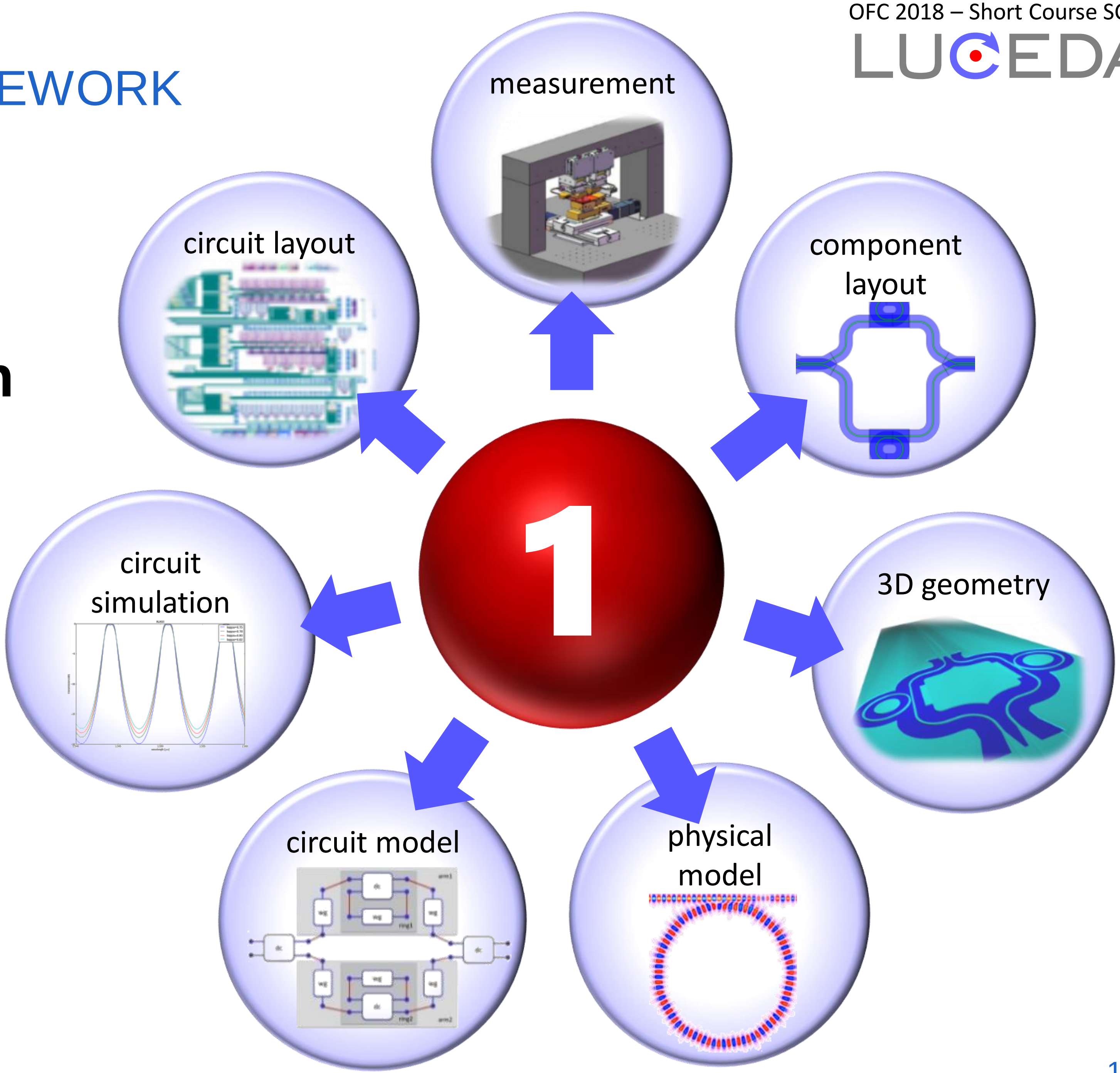
One component definition

for

Circuit design

Layout

Simulation

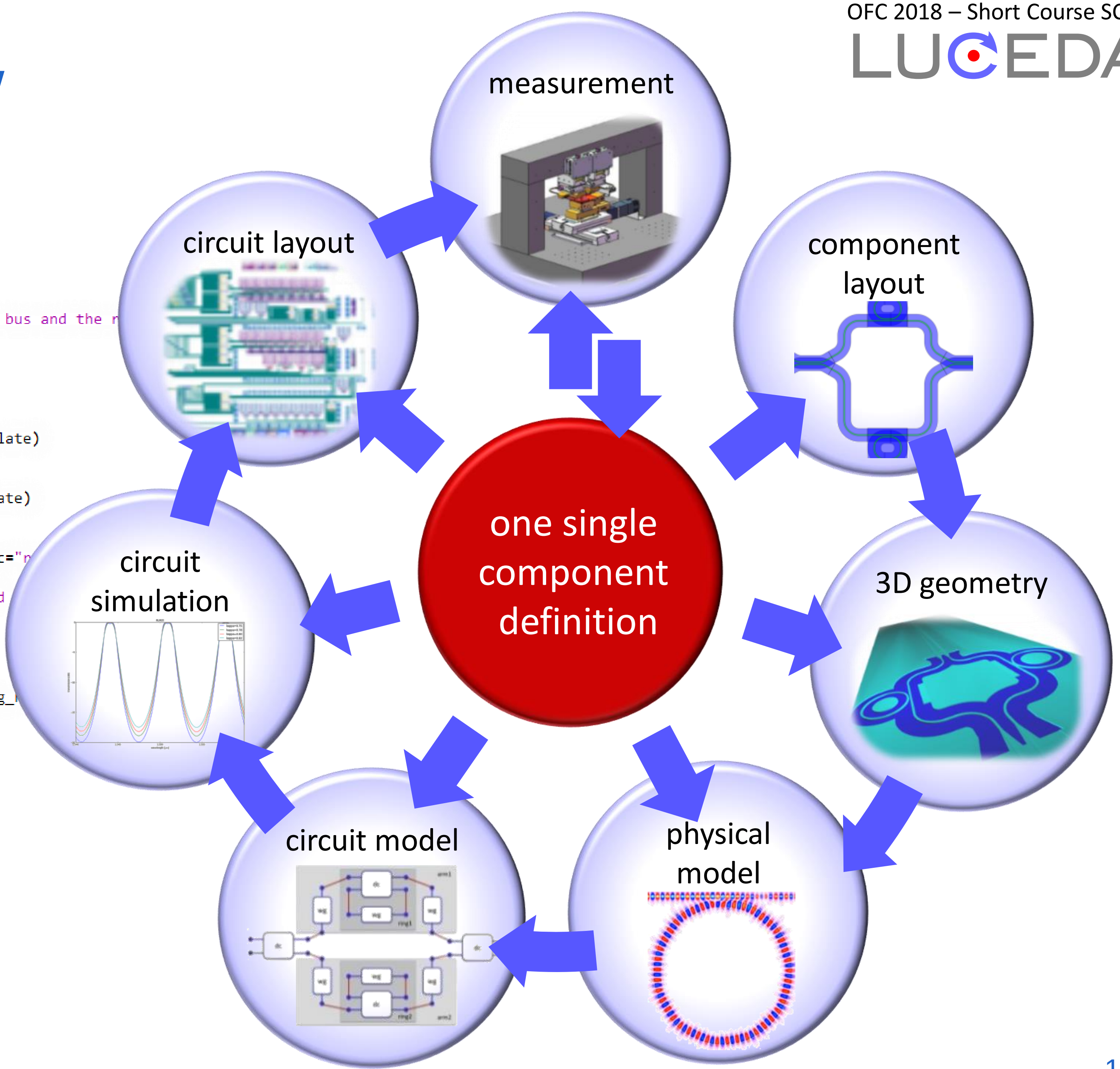


LUCEDA

The IPKISS Design Flow

Python script based

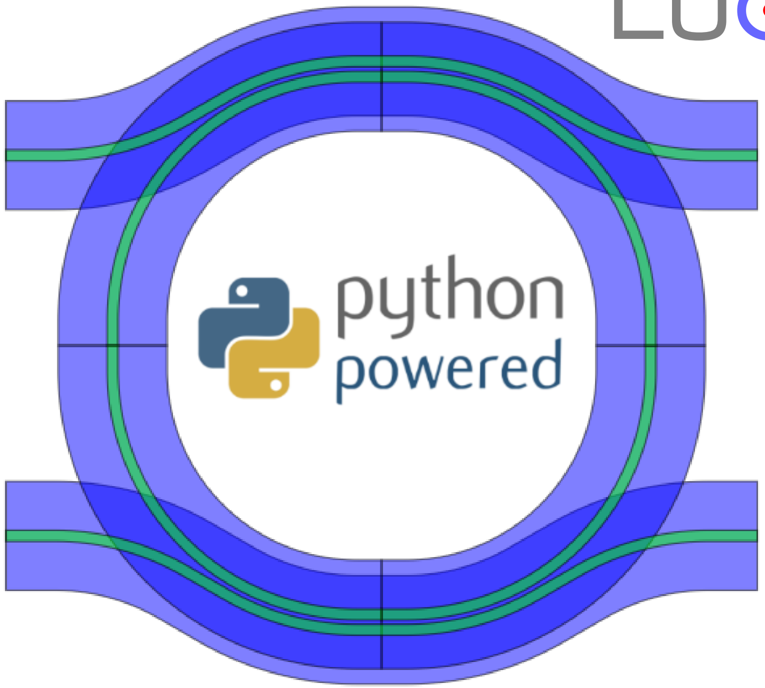
```
class RingResonator(i3.PCell):
    """A generic ring resonator class."""

    wg_template = i3.WaveguideTemplateProperty(default=TECH.PCELLS.WG.DEFAULT,
                                               doc="trace template used for the bus and the r

    bus = i3.ChildCellProperty(doc="bus waveguide")
    ring = i3.ChildCellProperty(doc="ring waveguide")

    def _default_ring(self):
        return i3.Waveguide(name=self.name+"_ring", trace_template=self.wg_template)

    def _defaul_bus(self):
        return i3.Waveguide(name=self.name+"_bus", trace_template=self.wg_template)

    class Layout(i3.LayoutView):
        ring_radius = i3.PositiveNumberProperty(default=TECH.WG.BEND_RADIUS, doc="r
        coupler_spacing = i3.PositiveNumberProperty(default=TECH.WG.DC_SPACING,
                                                    doc="spacing between bus and

        def _default_ring(self):
            ring_layout = self.cell.ring.get_default_view(i3.LayoutView)
            ring_layout.set(trace_template=self.wg_template,
                            shape=i3.ShapeCircle(center=(0, 0), radius=self.ring_
            return ring_layout

        def _default_bus(self):
            r, s = self.ring_radius, self.coupler_spacing
            bus_layout = self.cell.bus.get_default_view(i3.LayoutView)
            bus_layout.set(trace_template=self.wg_template,
                           shape=[(-r, -r-s), (+r, -r-s)])
            return bus_layout

        def _generate_instances(self, insts):
            insts += i3.SRef(name="ring", reference=self.ring)
            insts += i3.SRef(name="bus", reference=self.bus)
            return insts

        def _generate_ports(self, ports):
            ports += self.instances["bus"].ports
            return ports
```



THE IPKISS DESIGN FLOW

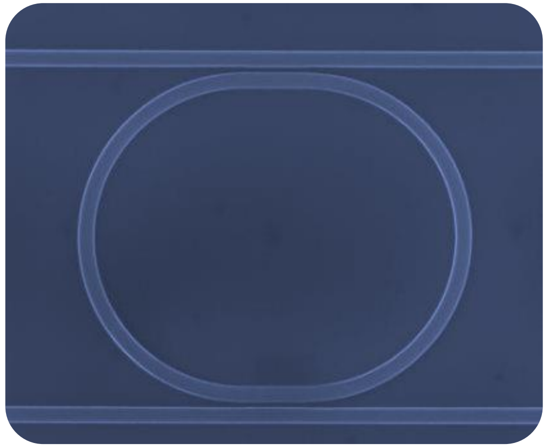
Python script based

```
class RingResonator(i3.PCell):
    """A generic ring resonator class."""

    wg_template = i3.WaveguideTemplateProperty(default=TECH.PCELLS.WG.DEFAULT,
                                               doc="trace template used for the bus and the ring")

    bus = i3.ChildCellProperty(doc="bus waveguide")
    ring = i3.ChildCellProperty(doc="ring waveguide")

    def _default_ring(self):
        return i3.Waveguide(name=self.name+"_ring", trace_template=self.wg_template)

    def _default_bus(self):
        return i3.Waveguide(name=self.name+"_bus", trace_template=self.wg_template)

    class Layout(i3.LayoutView):
        ring_radius = i3.PositiveNumberProperty(default=TECH.WG.BEND_RADIUS, doc="radius of ring")
        coupler_spacing = i3.PositiveNumberProperty(default=TECH.WG.DC_SPACING,
                                                    doc="spacing between bus and ring waveguide")

        def _default_ring(self):
            ring_layout = self.cell.ring.get_default_view(i3.LayoutView)
            ring_layout.set(trace_template=self.wg_template,
                            shape=i3.ShapeCircle(center=(0, 0), radius=self.ring_radius))
            return ring_layout

        def _default_bus(self):
            r, s = self.ring_radius, self.coupler_spacing
            bus_layout = self.cell.bus.get_default_view(i3.LayoutView)
            bus_layout.set(trace_template=self.wg_template,
                           shape=[(-r, -r-s), (+r, -r-s)])
            return bus_layout

        def _generate_instances(self, insts):
            insts += i3.SRef(name="ring", reference=self.ring)
            insts += i3.SRef(name="bus", reference=self.bus)
            return insts

        def _generate_ports(self, ports):
            ports += self.instances["bus"].ports
            return ports
```



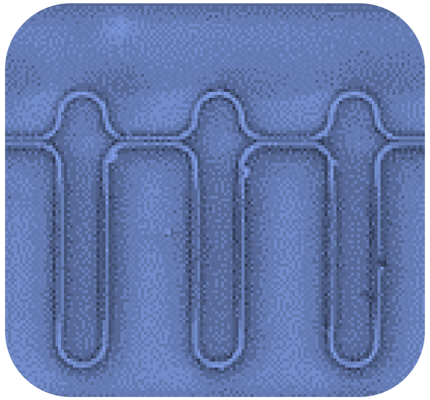
- extremely flexible
- easy-to-read
- powerful engineering libraries
- industry standard

THE PICAZZO LIBRARY



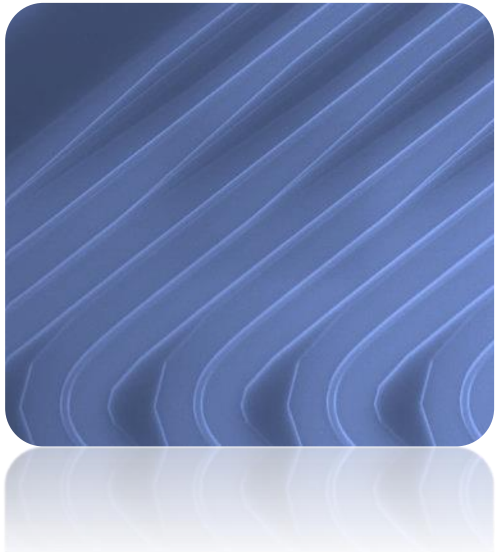
A large library of photonic components

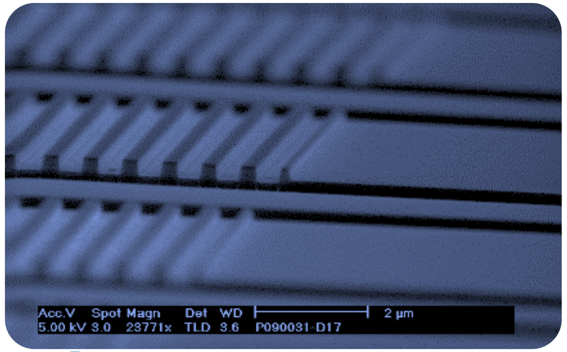
- waveguides and routing
- crossings, splitters and couplers
- wavelength filters
- grating couplers and mode converters
- generic modulator blocks

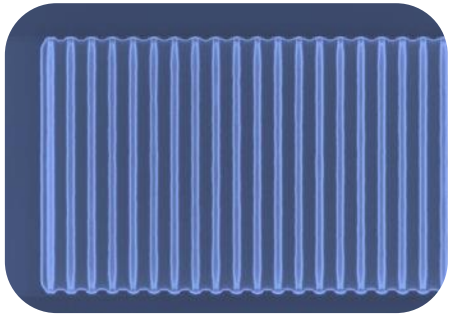


Parametric and technology aware

Validated on the IMEC technology platform







149

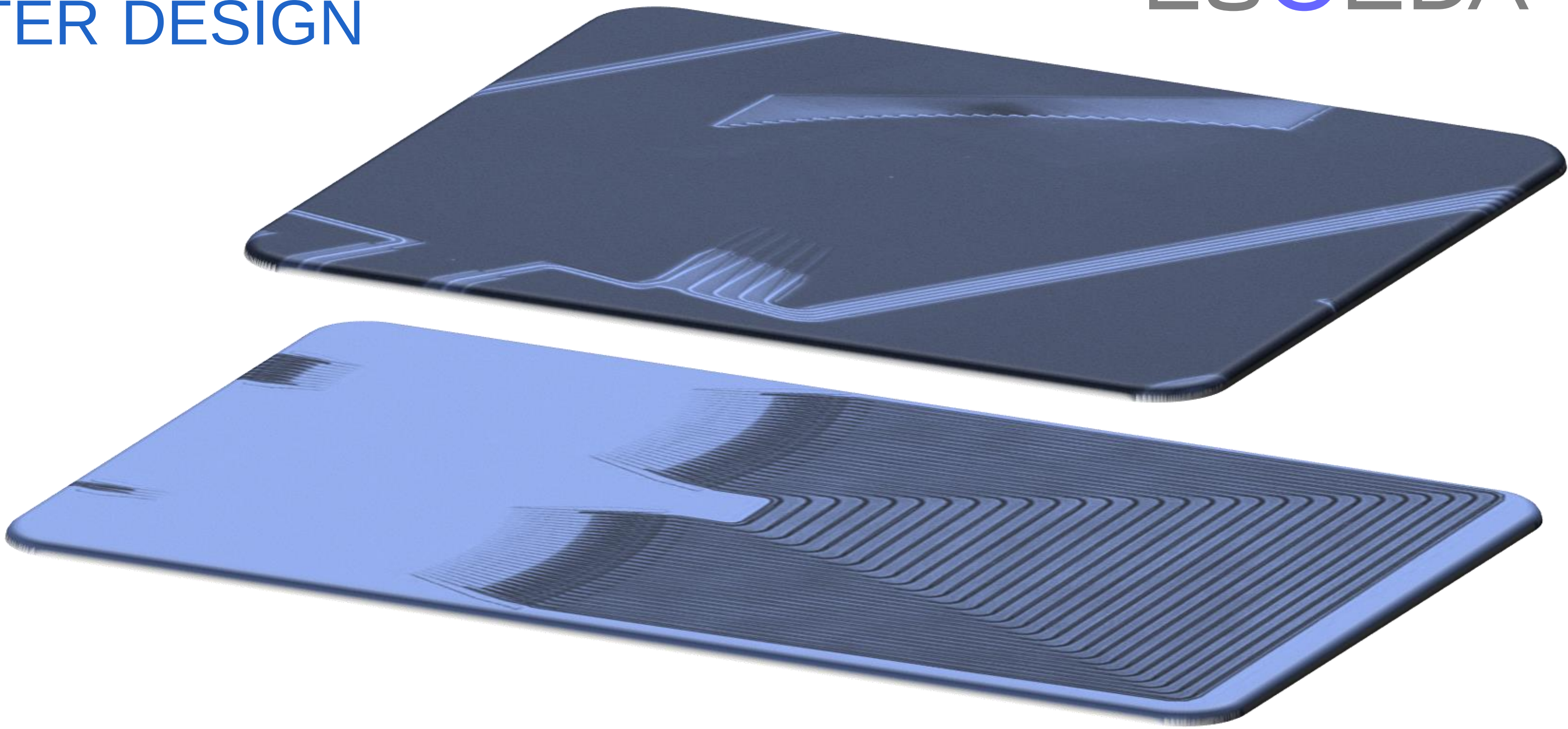
LUCEDA

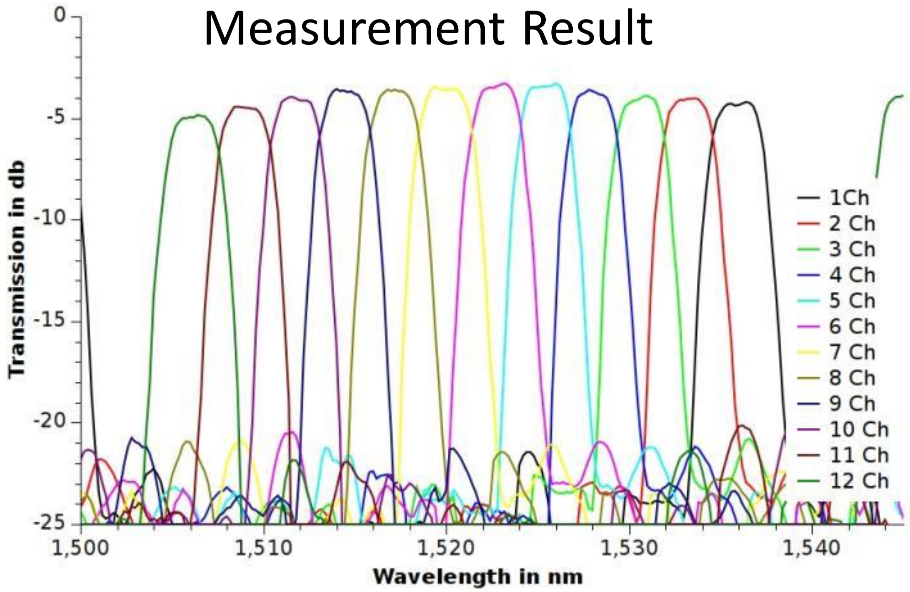
ADVANCED SPECTRAL FILTER DESIGN

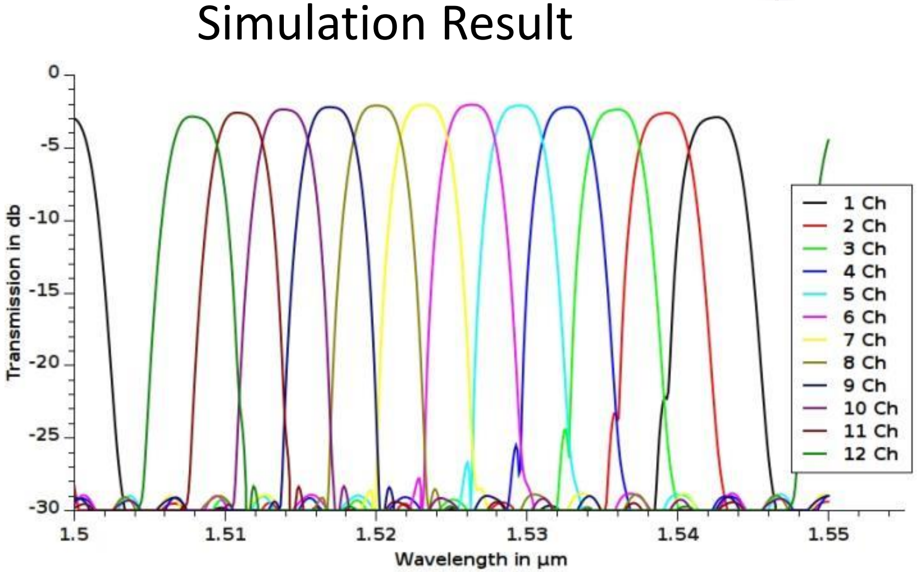
Arrayed Waveguide Gratings

Echelle Gratings

- Fully parametric
- Design from specifications
- Integrated layout and simulation
- Validated on fabricated devices







GHENT UNIVERSITY

imec

LUCEDA

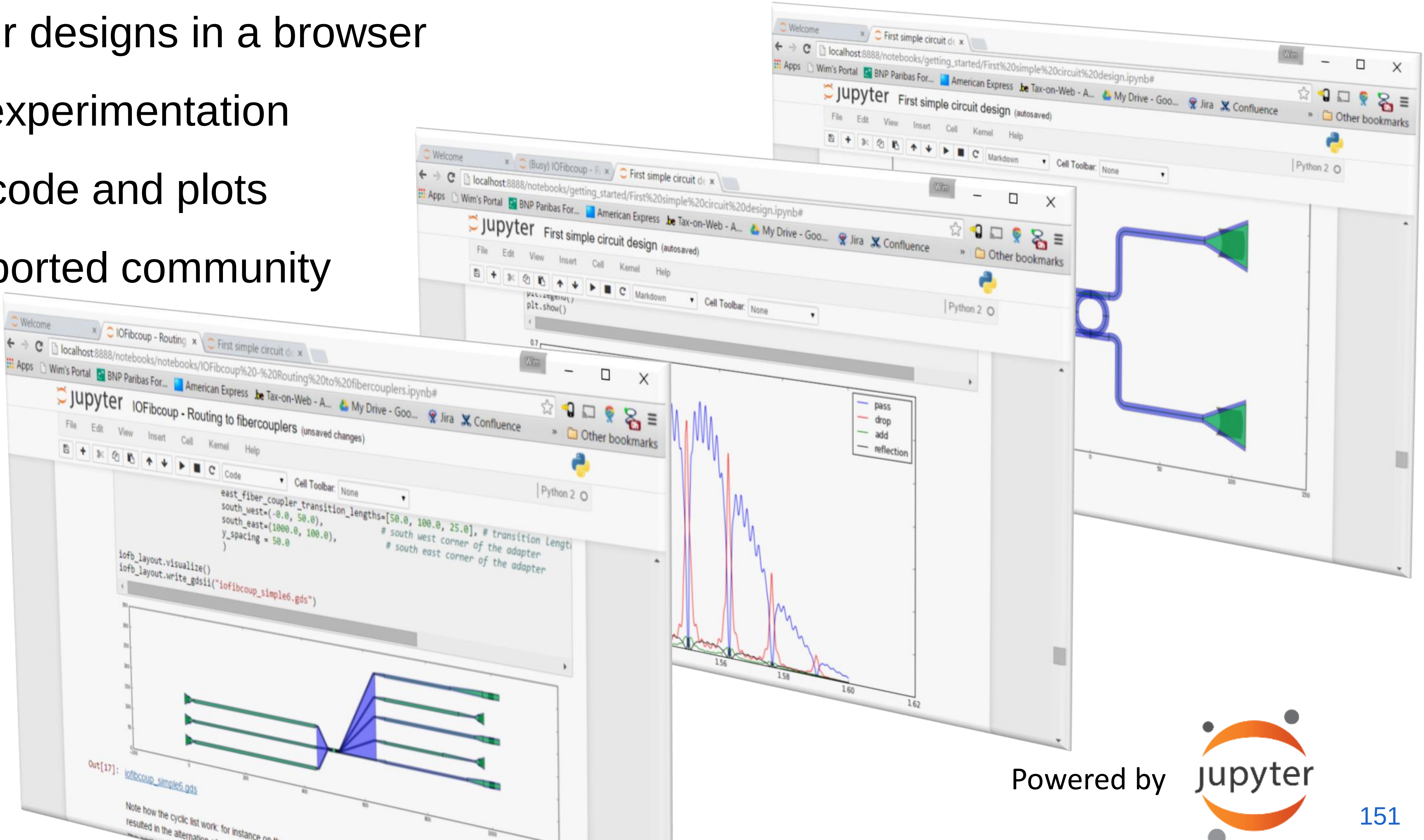
IPKISS Notebooks

Explore your designs in a browser

Very rapid experimentation

Interactive code and plots

Widely supported community



Powered by jupyter

GHENT UNIVERSITY

imec

FIRST NOTEBOOKS

Unfamiliar with Python?

/0_1_python_getting_started: *basic Python tutorial*

/0_2_ numpy_and_plotting: *Numpy and Matplotlib*

Check if everything works and if you find your way around the notebook interface.

PRACTICAL

1. Connect WIFI / Ethernet
2. Open web browser (Chrome, Firefox, Opera)
3. Connect to Jupyter server
   (address will be provided on-site)
4. Log in with your personal ID/password

NOTEBOOK: INTERACTIVE ENVIRONMENT

Text and explanations

Variables

A name that is used to denote something or a value is called a variable. In python, variables can be declared and values can be assigned to it as follows,

In [2]:
```
x = 2
y = 5
xy = 'Hey'
```

In [3]:
```
print x+y, xy
```

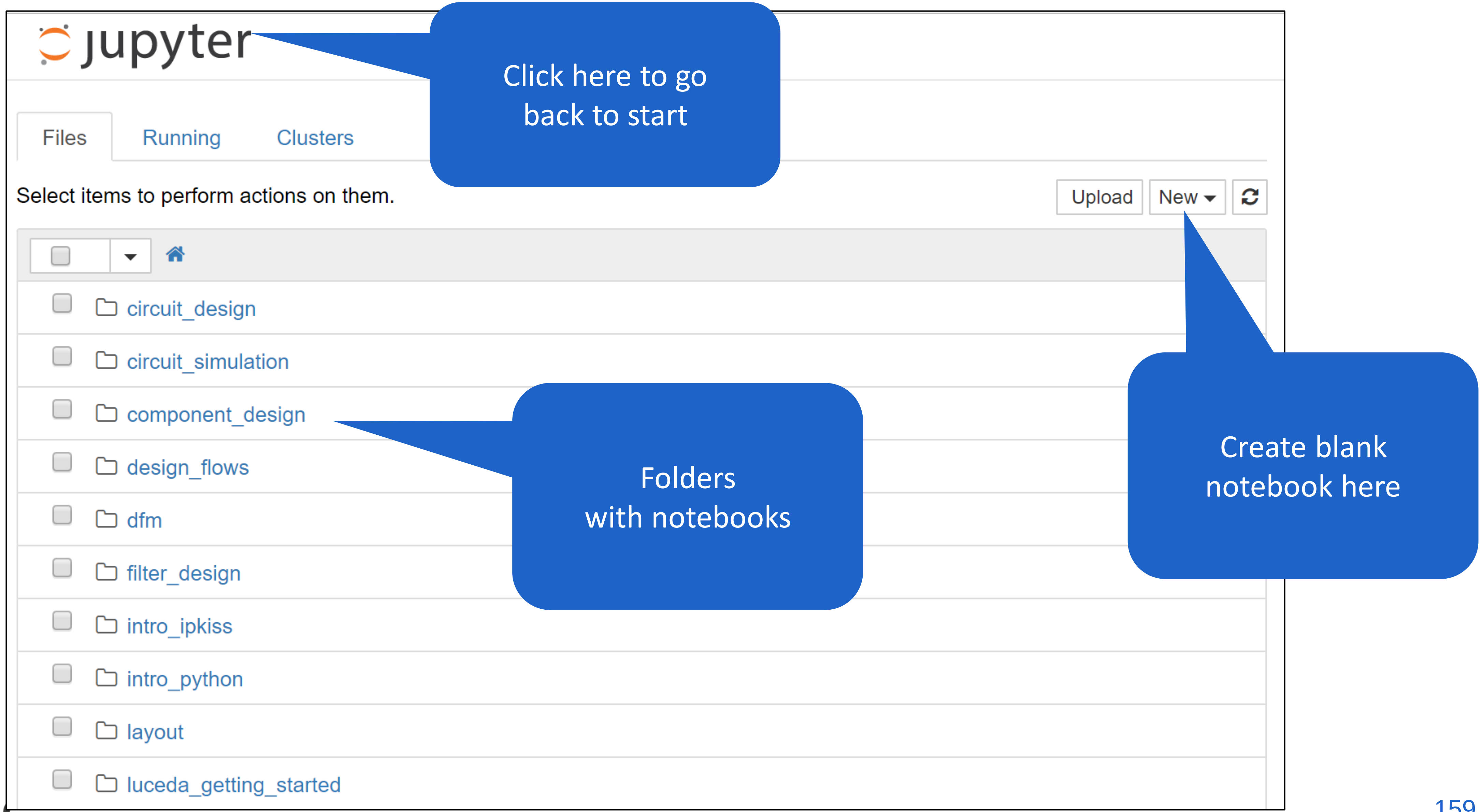
```
7 Hey
```

Executable python code

SHIFT+ENTER

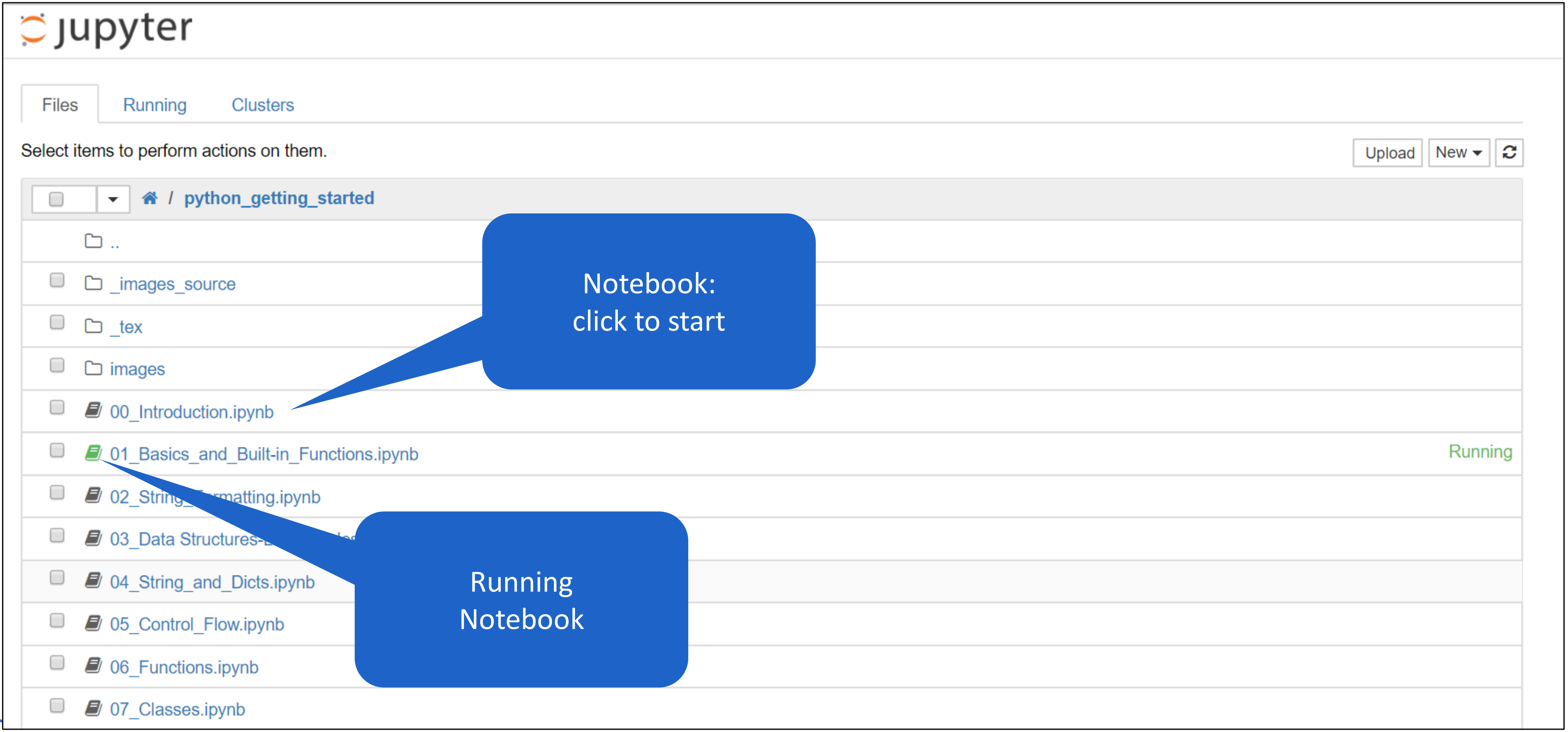
to execute

NAVIGATING



GHENT UNIVERSITY
imec

NAVIGATING

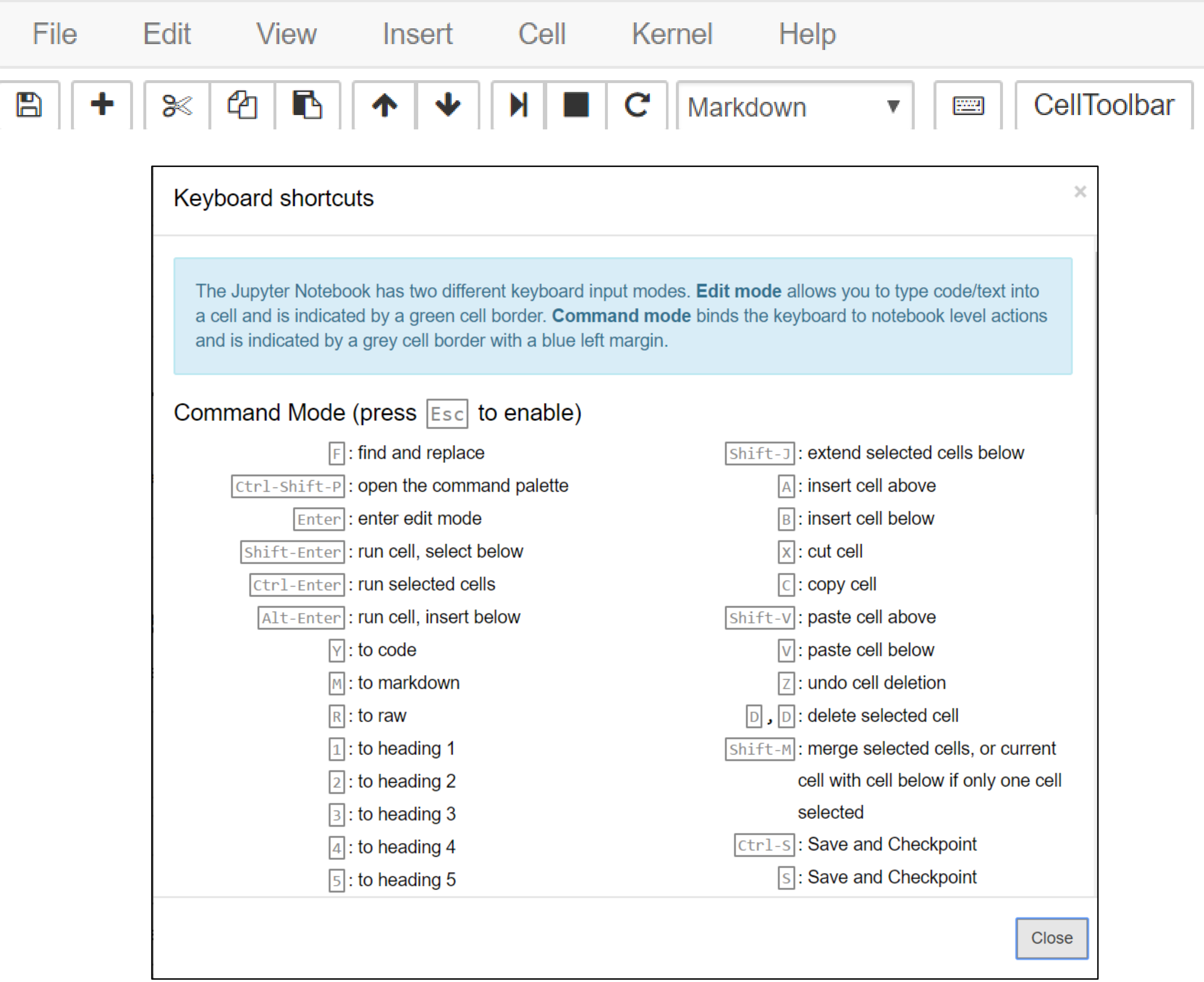

jupyter
Files
Running
Clusters
Select items to perform actions on them.
Upload
New
python_getting_started
..
_images_source
_tex
images
00_Introduction.ipynb
01_Basics_and_Built-in_Functions.ipynb
Running
04_String_and_Dicts.ipynb
05_Control_Flow.ipynb
06_Functions.ipynb
07_Classes.ipynb
Notebook:
click to start
Running
Notebook

PRESS H FOR 'HELP'

Useful menu and toolbar

Keyboard shortcuts

are extremely powerful



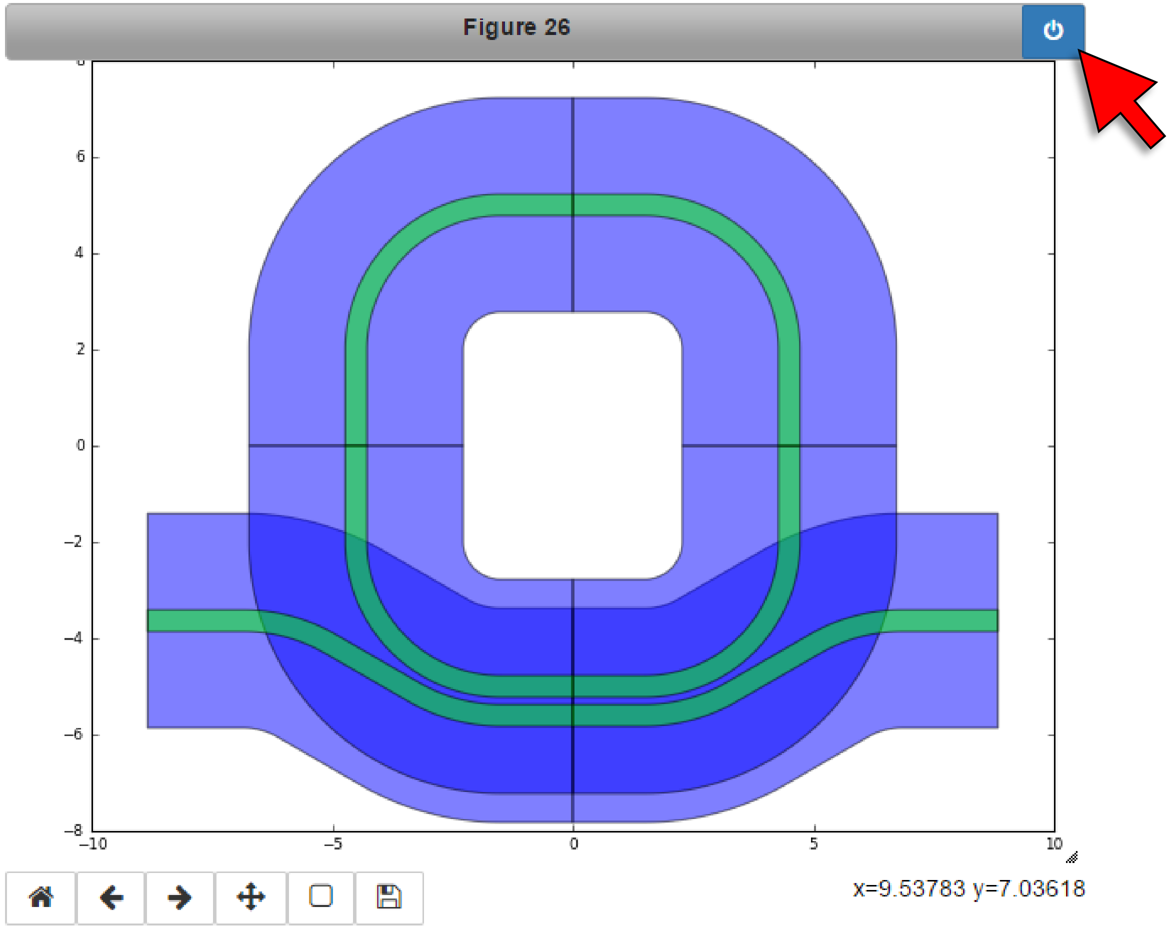
GHENT UNIVERSITY imec

Take care of Memory

Interactive plots consume resources.

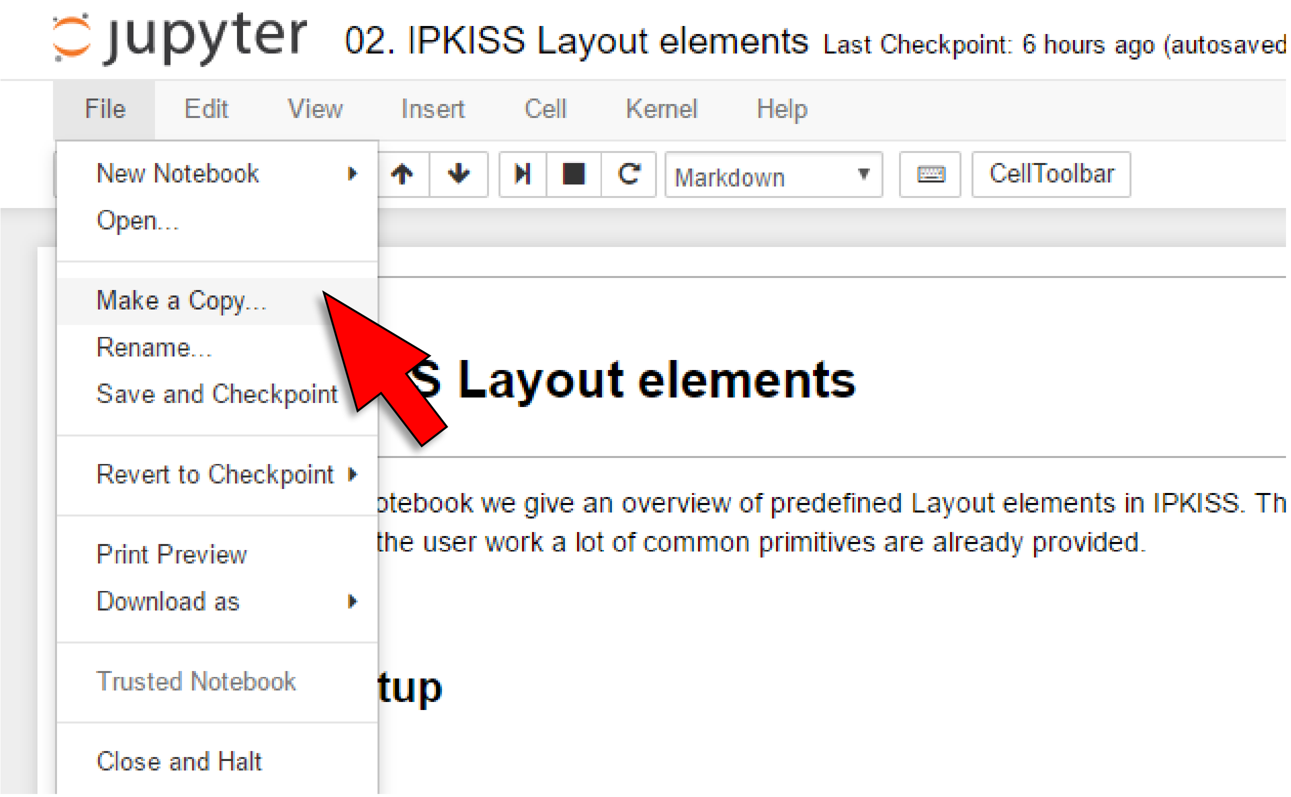
Close them when ready.

```
C:\luceda\ipkiss_311\python\envs\ipkiss3\lib\site-packages\matplotlib\pyplot.py:516: RuntimeWarning: More than 20 figures have
 been opened. Figures created through the pyplot interface (`matplotlib.pyplot.figure`) are retained until explicitly closed an
d may consume too much memory. (To control this warning, see the rcParam `figure.max_open_warning`).
  max_open_warning, RuntimeWarning)
```



GETTING STARTED…

- connect to the internet
- open browser (Chrome, Firefox)
- connect to notebook server: https://wsjupyler.intec.ugent.be
- notebook login / password



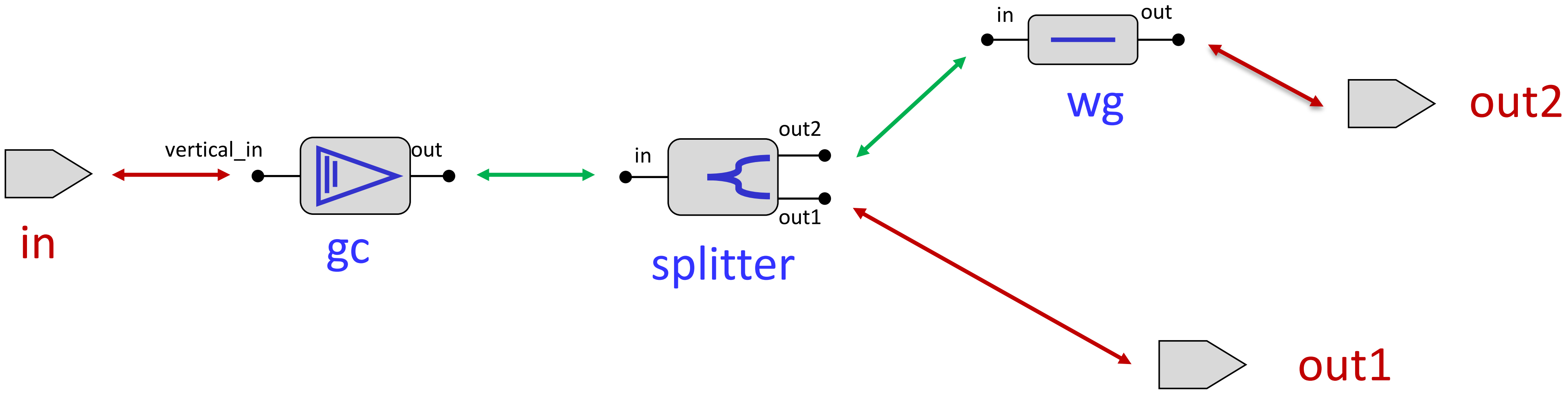
Launch a notebook

Step 1:
Copy the notebook

BUILDING CIRCUITS IN A NOTEBOOK

Define schematics in python code

- List building blocks (or subcircuits)
  - gc, splitter, wg
- List internal connections
  - gc:out↔splitter:in, splitter:out2↔wg:in
- List external ports
  - in ↔gc:vertical_in, out1 ↔ splitter:out1, out2 ↔wg:out



Building Circuits: AutoPlaceAndConnect

Circuits with direct connections: no waveguide generation

```
from addon_luceda.auto_place_and_connect import AutoPlaceAndConnect

child_cells = {"dc1": my_dircoup,
               "dc2": my_dircoup,
               "wg1": my_wg,
               "wg2": my_wg}

links = [("dc1:out2", "wg1:in"),
         ("wg1:out" , "dc2:in2"),
         ("dc2:out2", "wg2:in"),
         ("wg2:out", "dc1:in2")]

external_port_names = {"dc1:in1"  : "in1",
                       "dc1:out1" : "out1",
                       "dc2:in1"  : "in2",
                       "dc2:out1" : "out2"}

my_ring = AutoPlaceAndConnect(child_cells=child_cells,
                              links=links,
                              external_port_names=external_port_names)
my_ring_lo = my_ring.Layout()
my_ring_lo.visualize(annotate=True)
```

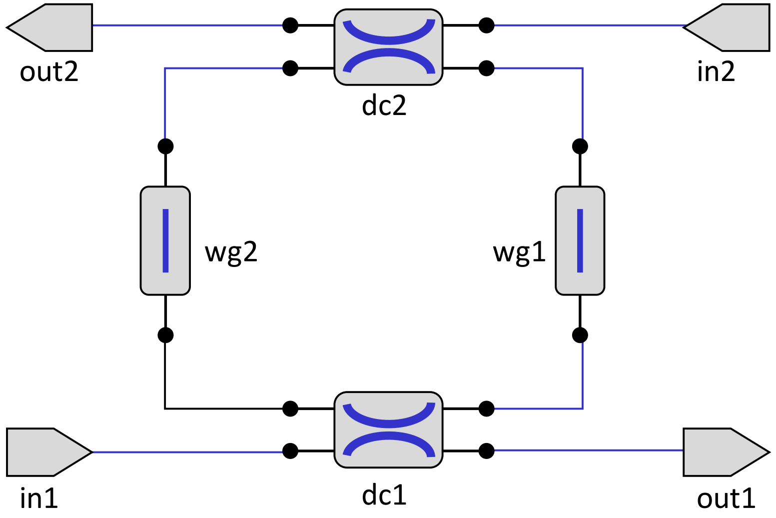
4 components

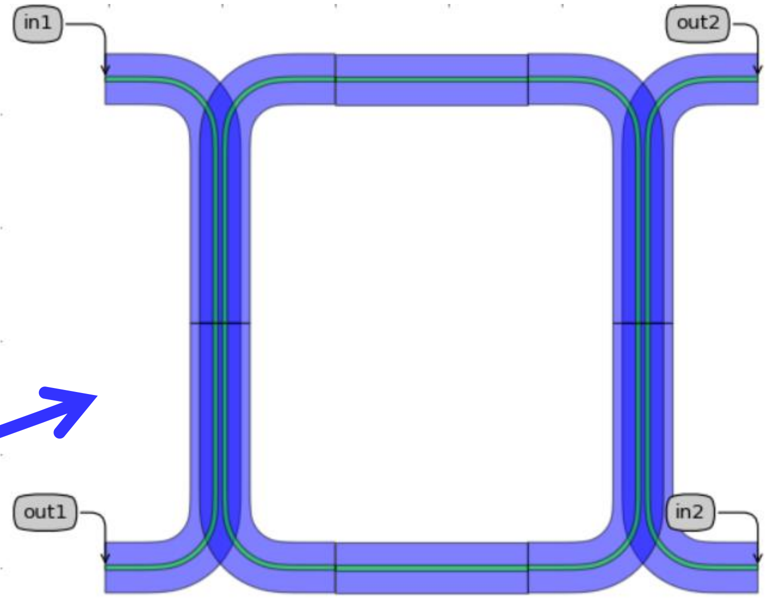
4 internal connections

4 input/output ports

automatic placement

auto-generate layout



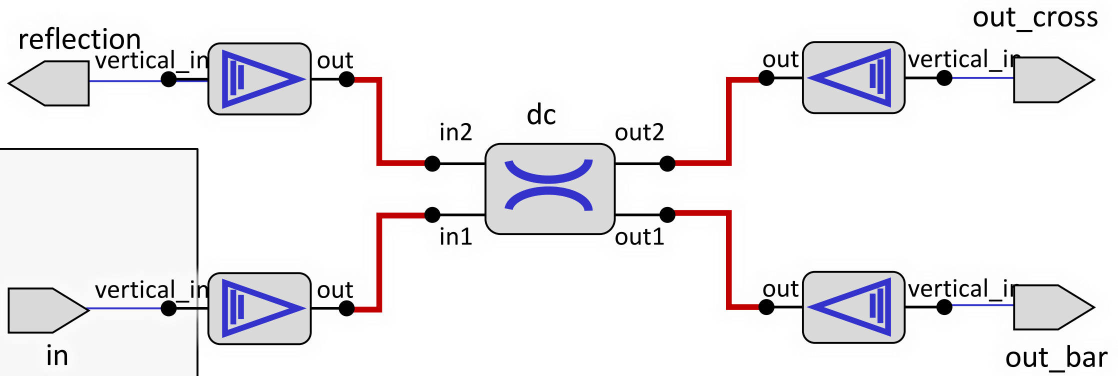


GHENT UNIVERSITY

imec

BUILDING CIRCUITS: PLACEANDAUTOROUTE

Generate waveguides for connections




```
from picazzo3.routing.place_route import PlaceAndAutoRoute

dc_circuit = PlaceAndAutoRoute(name="dc_with_gc",
                  child_cells={"dc": my_dc,
                               "gc_in" : fc,
                               "gc_out_bar" : fc,
                               "gc_out_cross" : fc,
                               "gc_reflection" : fc
                              },
                  links=[("gc_in:out", "dc:in1"),
                         ("gc_reflection:out", "dc:in2"),
                         ("dc:out1", "gc_out_bar:out"),
                         ("dc:out2", "gc_out_cross:out"),
                         ],
                  external_port_names={"gc_in:vertical_in": "in",
                                       "gc_out_bar:vertical_in": "out_bar",
                                       "gc_out_cross:vertical_in": "out_cross",
                                       "gc_reflection:vertical_in": "reflection"}
                  )

transformations = {"gc_in": i3.Translation((-100,-20)),
                   "gc_out_cross": i3.Rotation(rotation=180) + i3.Translation((100, +20)),
                   "gc_out_bar": i3.Rotation(rotation=180) + i3.Translation((100, -20)),
                   "gc_reflection": i3.Rotation(rotation=180) + i3.Translation((100, +60)),
                   }

dc_circuit_layout = dc_circuit.Layout(child_transformations=transformations,
                   bend_radius=10.0)
dc_circuit_layout.visualize(annotate=True)
```

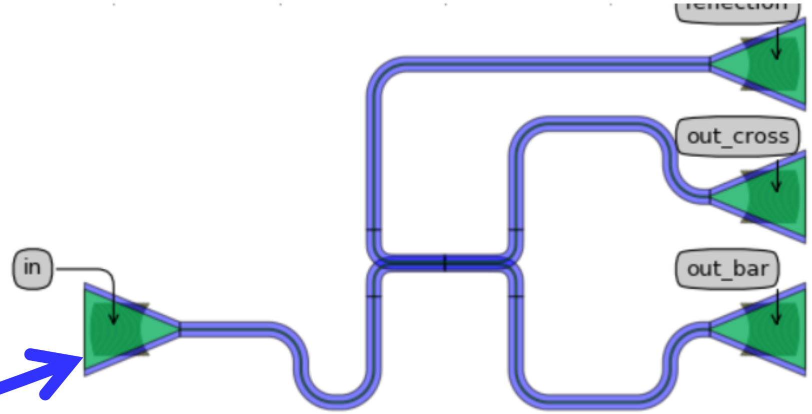
5 components

4 internal connections

4 input/output ports

manual placement

auto-generate layout

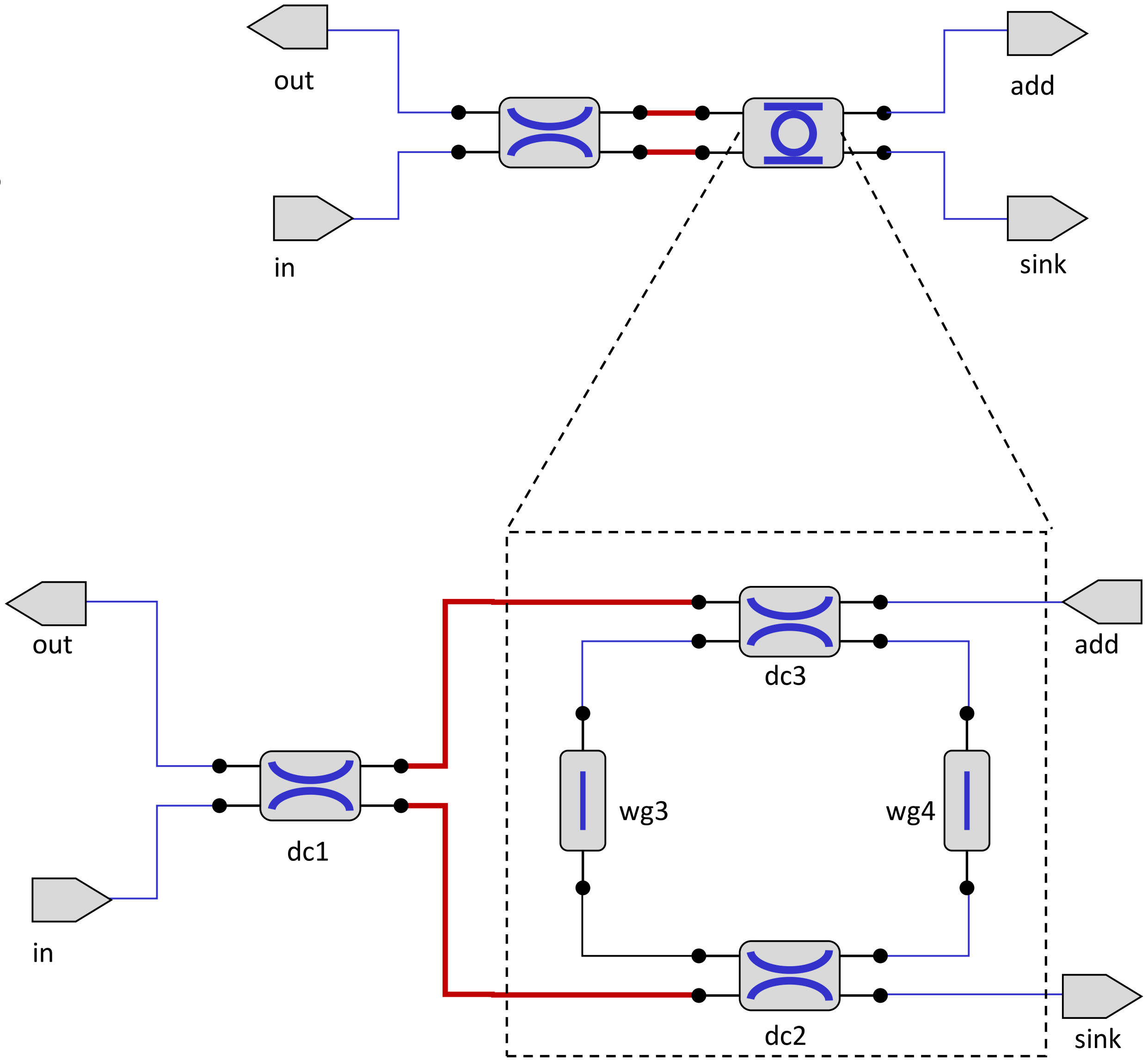


GHENT UNIVERSITY
imec

USE HIERARCHY: YOU CAN USE A CIRCUIT AS A BUILDING BLOCK

Circuits can be nested

Break up circuits into reusable parts



GHENT UNIVERSITY imec

THE SMALL PRINT ON COPYRIGHT

The material on the server is copyrighted

- The IPKISS toolset
- The addon libraries
- The notebooks

Please do not download the material to your own PC. It will probably not work as the server has a specific set of pre-configured utilities.

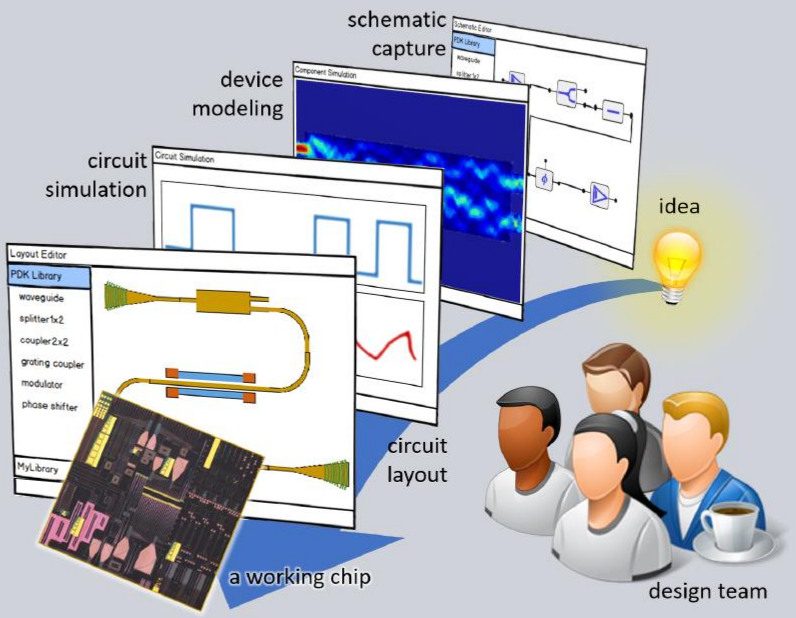
If you interested in using IPKISS, contact info@lucedaphotonics.com

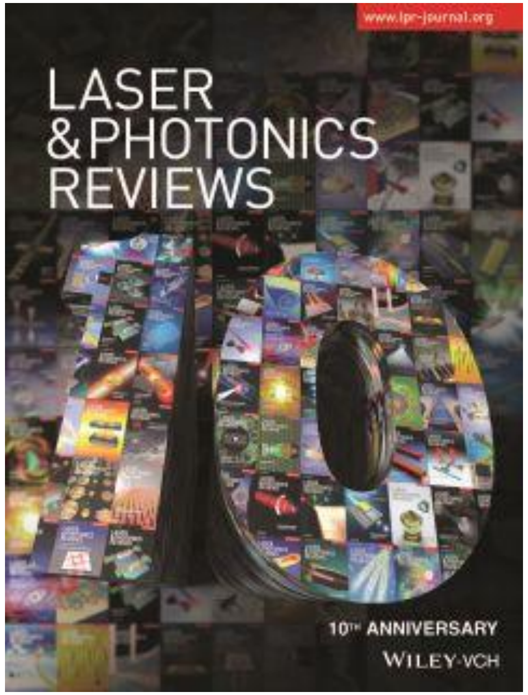
If you are interested in using the course material, contact wim.bogaerts@ugent.be

You can continue to use the server until 30 June 2018.

Further reading

Abstract Silicon Photonics technology is rapidly maturing as a platform for larger-scale photonic circuits. As a result, the associated design methodologies are also evolving from component-oriented design to a more circuit-oriented design flow, that makes abstraction from the very detailed geometry and enables design on a larger scale. In this paper, we review the state of this emerging photonic circuit design flow and its synergies with electronic design automation (EDA). We cover the design flow from schematic capture, circuit simulation, layout and verification. We discuss the similarities and the differences between photonic and electronic design, and the challenges and opportunities that present themselves in the new photonic design landscape, such as variability analysis, photonic-electronic co-simulation and compact model definition.

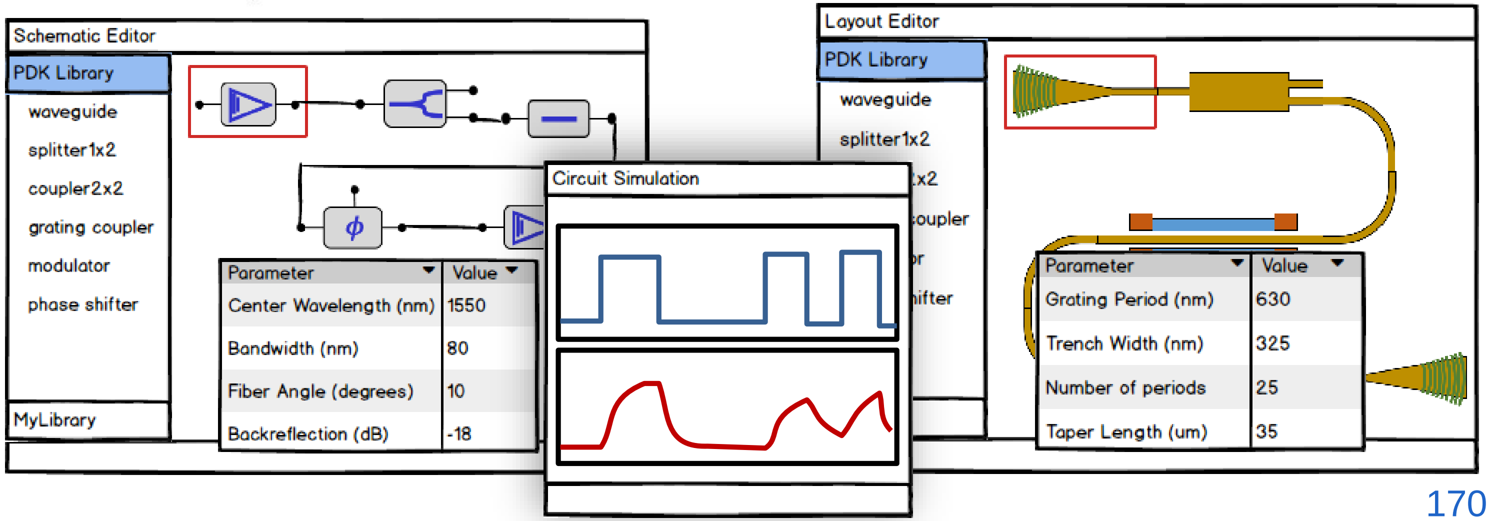




Silicon Photonics Circuit Design: Methods, Tools and Challenges (invited)

Wim Bogaerts[1,2,*] and Lukas Chrostowski[3]

Lasers and Photonics Reviews



GHENT UNIVERSITY imec

PHOTONICS RESEARCH GROUP

Wim Bogaerts

Professor in Silicon Photonics

E wim.bogaerts@ugent.be

T +32 9 264 3324

@PhotonicsUGent

www.photonics.intec.ugent.be

GHENT UNIVERSITY

imec embracing a better life

FACULTY OF ENGINEERING AND ARCHITECTURE

nb-photonics Center for Nano- and Biophotonics Ghent University

erc European Research Council