The New Golden Age of Open Silicon

Tim Edwards

VP Analog and Platform

efabless

efabless

efabless.com

Open Circuit Design

opencircuitdesign.com

November 7, 2019

WOSET2019 ICCAD

The New Golden Age of Open Silicon

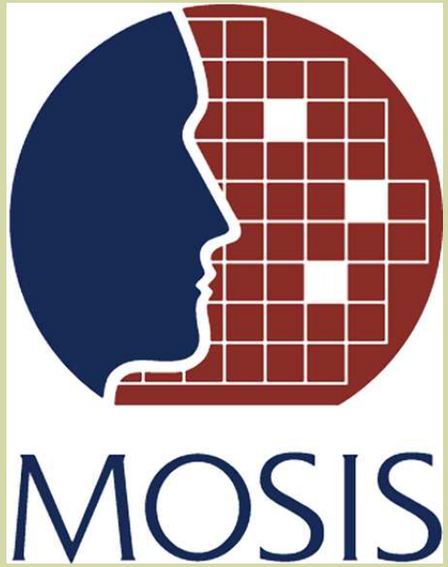
- A brief history of SCMOS
- A proposal for an open PDK Framework
- The New Golden Age of Open Silicon

A Brief History of SCMOS

SCMOS = Scalable CMOS

Once upon a time

in the 1980s. . .



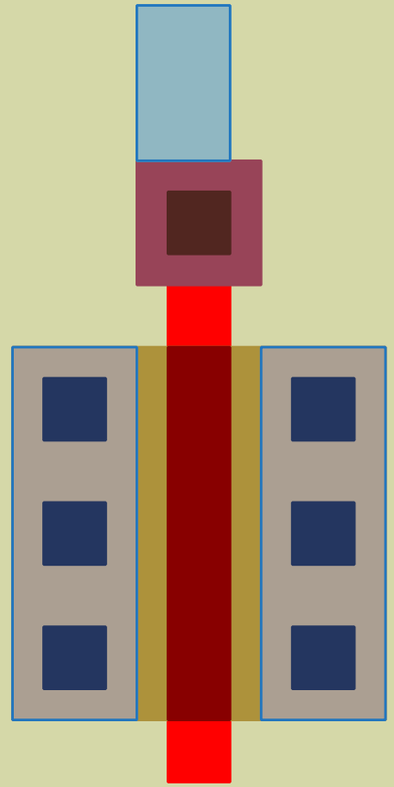
A Brief History of SCMOS

SCMOS = Scalable CMOS

grid size

λ = □

lambda (unitless)

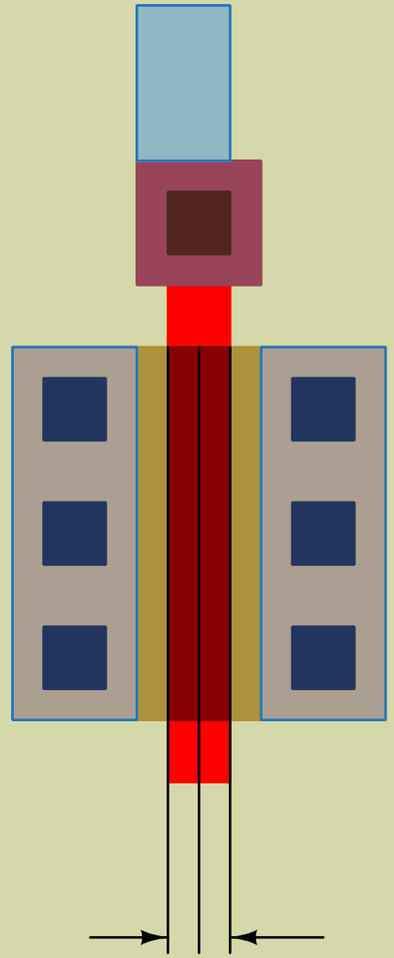


A Brief History of SCMOS

SCMOS = Scalable CMOS

λ = □

lambda (unitless)

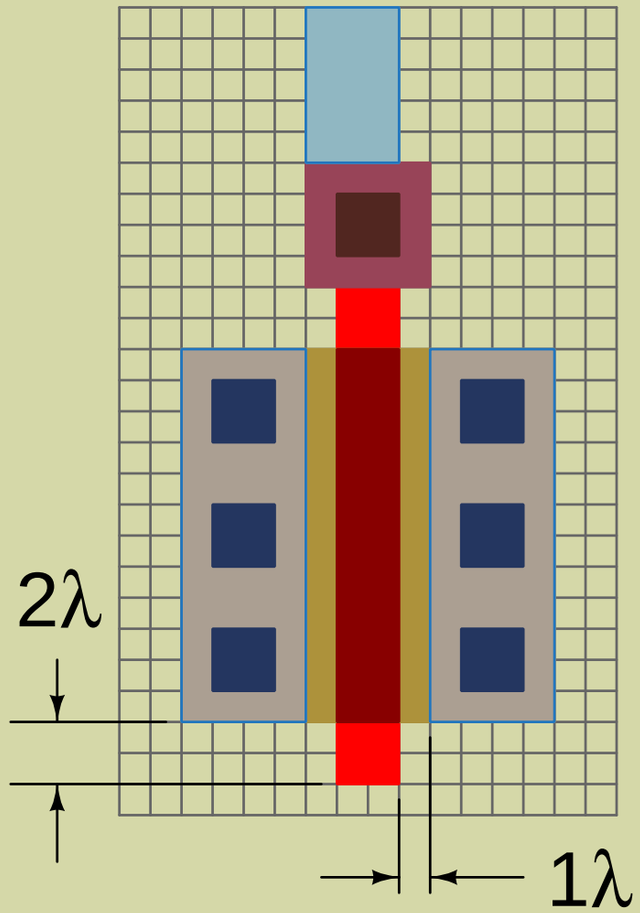


minimum gate length = 2λ

A Brief History of SCMOS

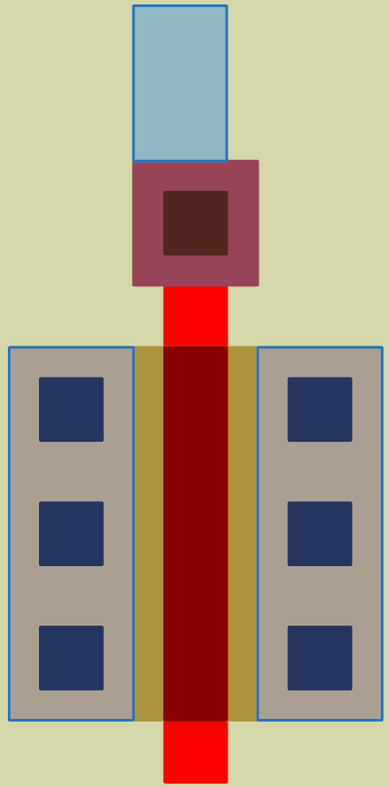
SCMOS = Scalable CMOS

- minimum gate length = 2λ
- minimum metal width = 3λ
- contact (cut + surround) width = 4λ
- poly extension of gate = 2λ
- poly to diffusion contact spacing = 1λ



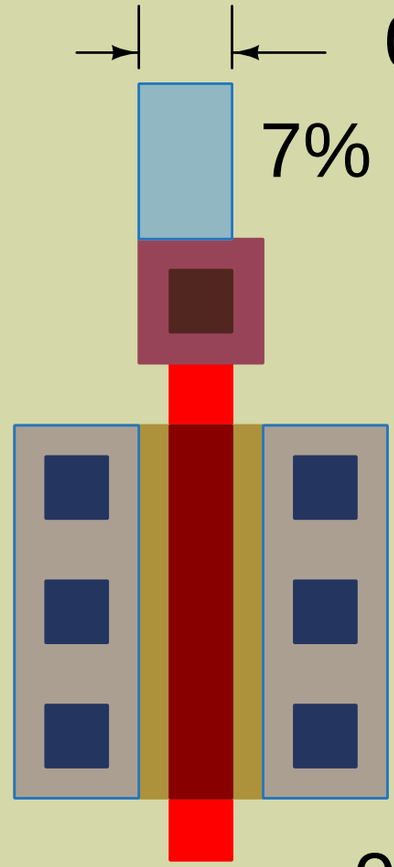
A Brief History of SCMOS

SCMOS dimensions map to foundry dimensions



0.18 µm process

output scale
λ = 0.1µm



0.3 µm

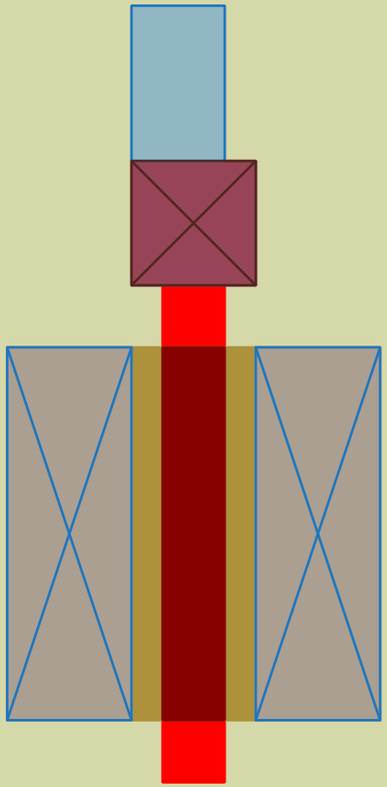
7% > minimum

0.2 µm

11% > minimum

A Brief History of SCMOS

layout

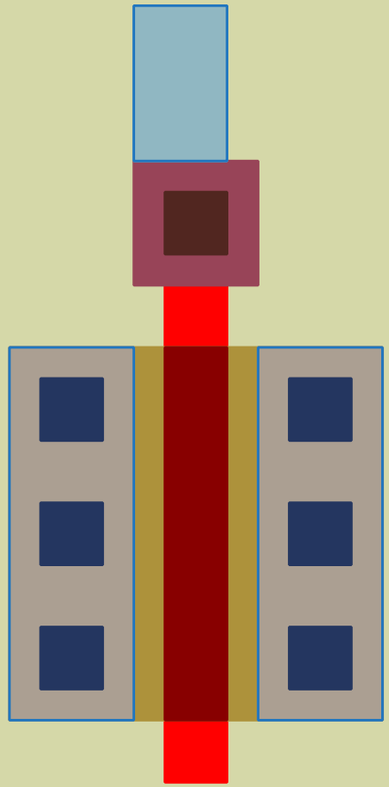


SCMOS simplifications

contacts represented by contact areas

cuts are generated by algorithm

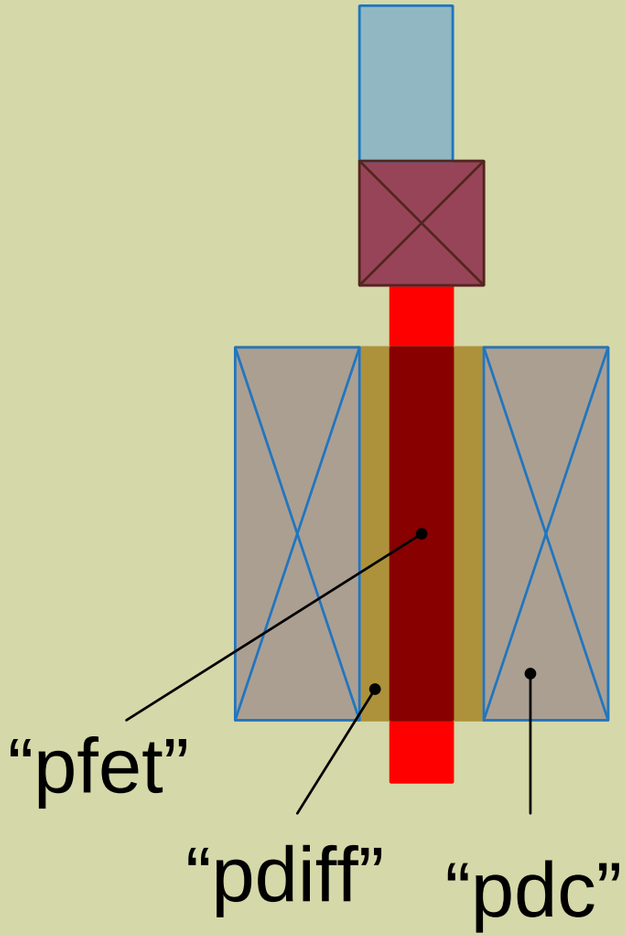
mask



“reinvented” as LEF
“VIARULE GENERATE”

A Brief History of SCMOS

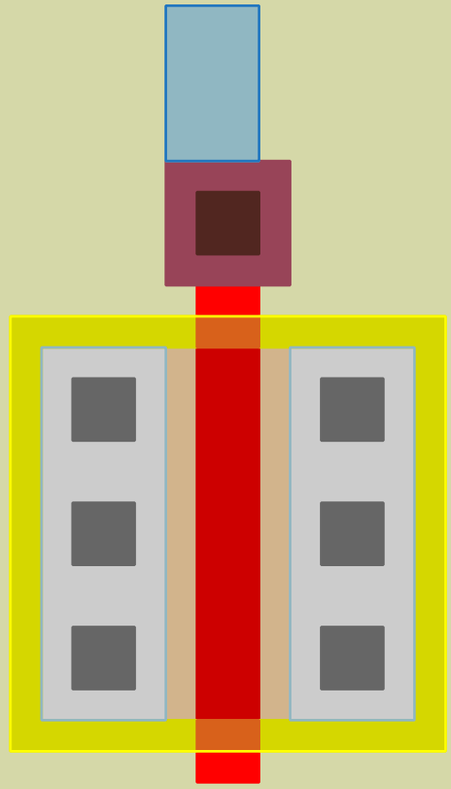
layout



SCMOS simplifications

Implant layers and sometimes well layers are inferred from drawn types and generated automatically in the output.

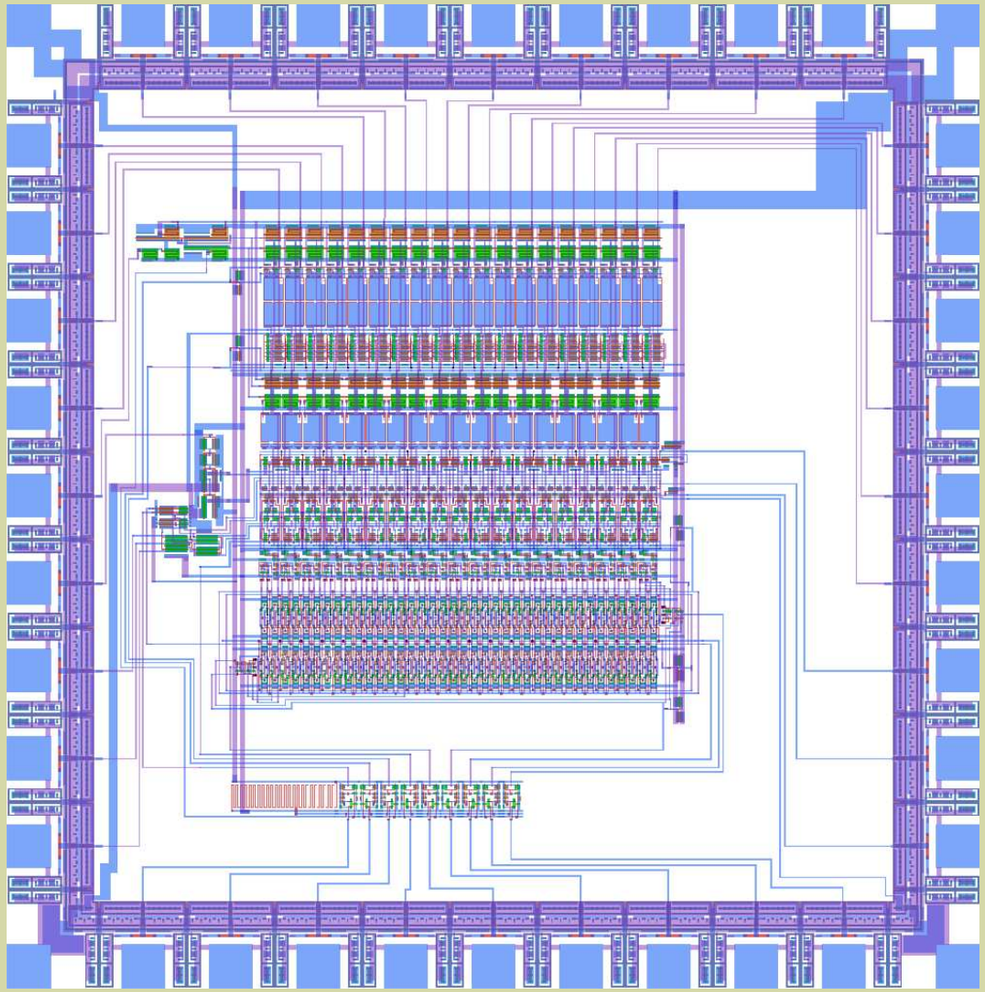
mask



PPLUS implant

A Brief History of SCMOS

One benefit of SCMOS: Long-term viability



SCMOS is forever:
A layout I did in 1997

A Brief History of SCMOS

Prime benefit of SCMOS: Natural obfuscation

Foundry rules cannot be reverse-engineered from SCMOS rules.

Layouts can be published and shared.

However. . .

Foundry rules *can* be reverse-engineered from the output generation mapping rules.

Solution: Services like MOSIS map SCMOS layout to foundry mask layout

A Brief History of SCMOS

Prime benefit of SCMOS: Natural obfuscation

Foundry rules cannot be reverse-engineered from SCMOS rules.

Layouts can be published and shared.

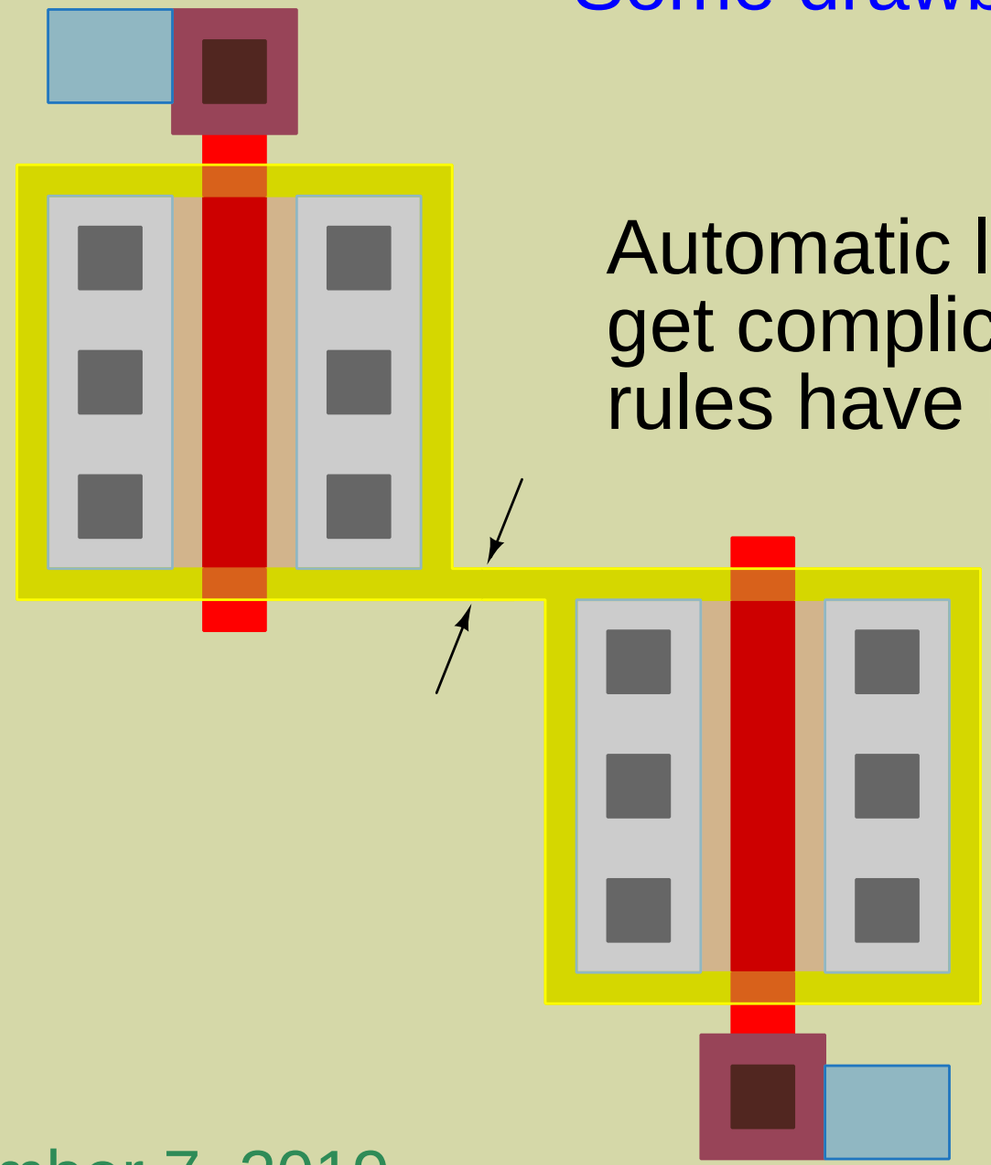
Also. . .

Cannot simulate without models, which are foundry-proprietary

Solution: Cloud-based simulation?

A Brief History of SCMOS

Some drawbacks of SCMOS



Automatic layer generation can get complicated. Simple generation rules have unintended consequences.

A Brief History of SCMOS

Some drawbacks of SCMOS

Scalable CMOS is not really scalable below about 0.25µm processes

Too many one-off DRC rules

Rules that don't map well to multiples of λ

But. . .

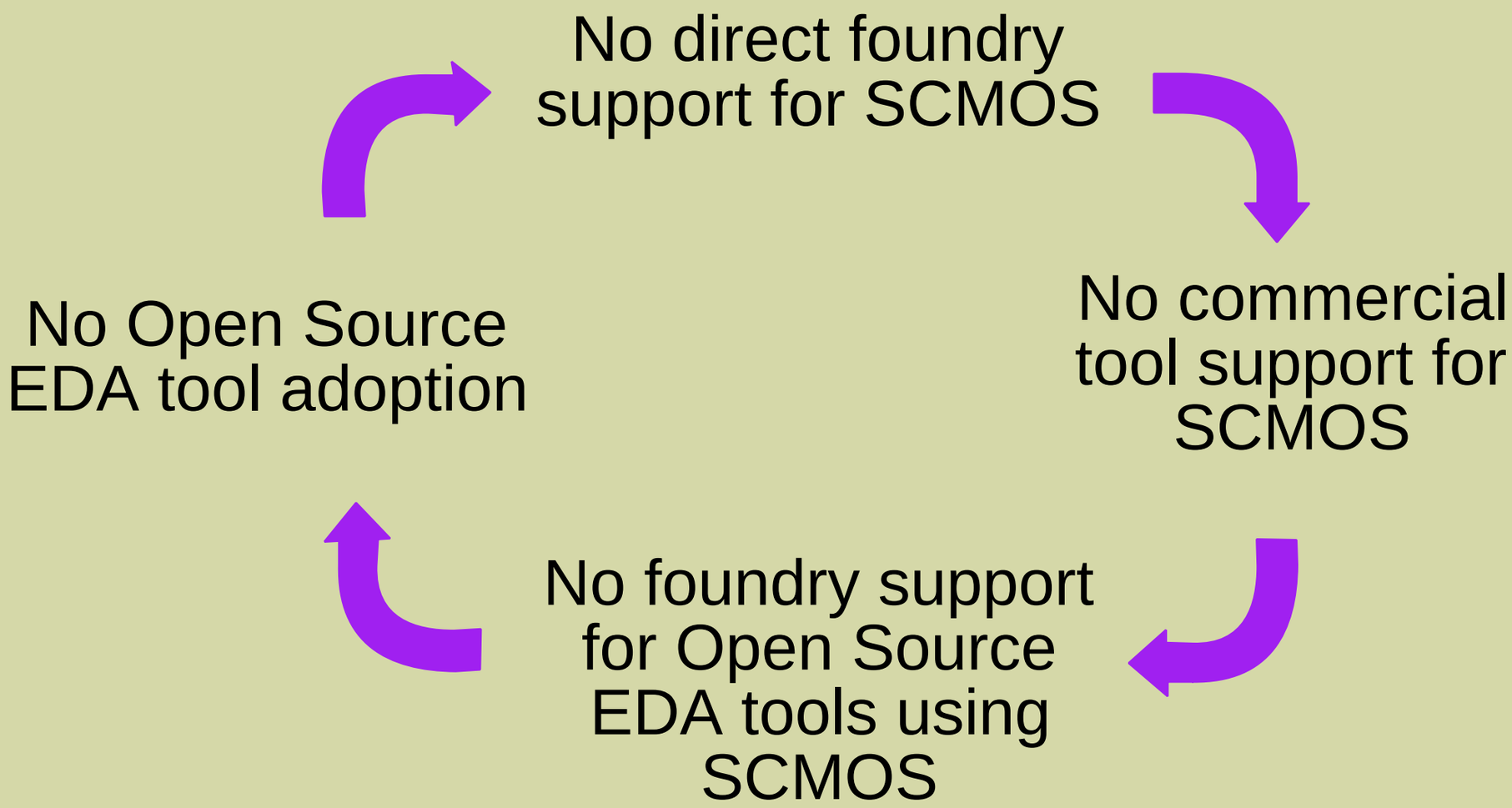
SCMOS rules work and simplify design
even if they cannot scale between processes.

The New Golden Age of Open Silicon

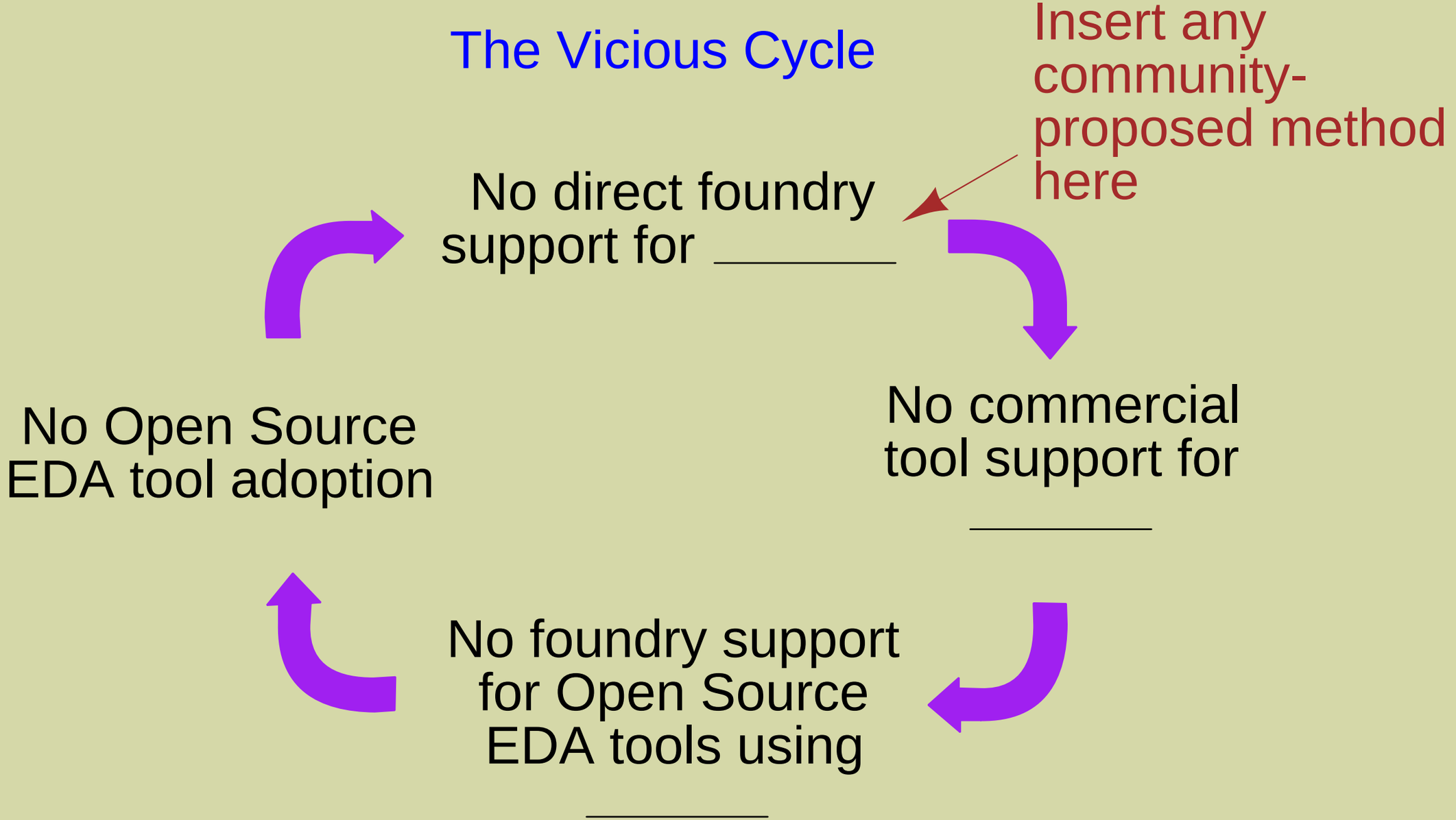
- A brief history of SCMOS
- A proposal for an open PDK Framework
- The New Golden Age of Open Silicon

A Proposal for an Open PDK Framework

The Vicious Cycle



A Proposal for an Open PDK Framework



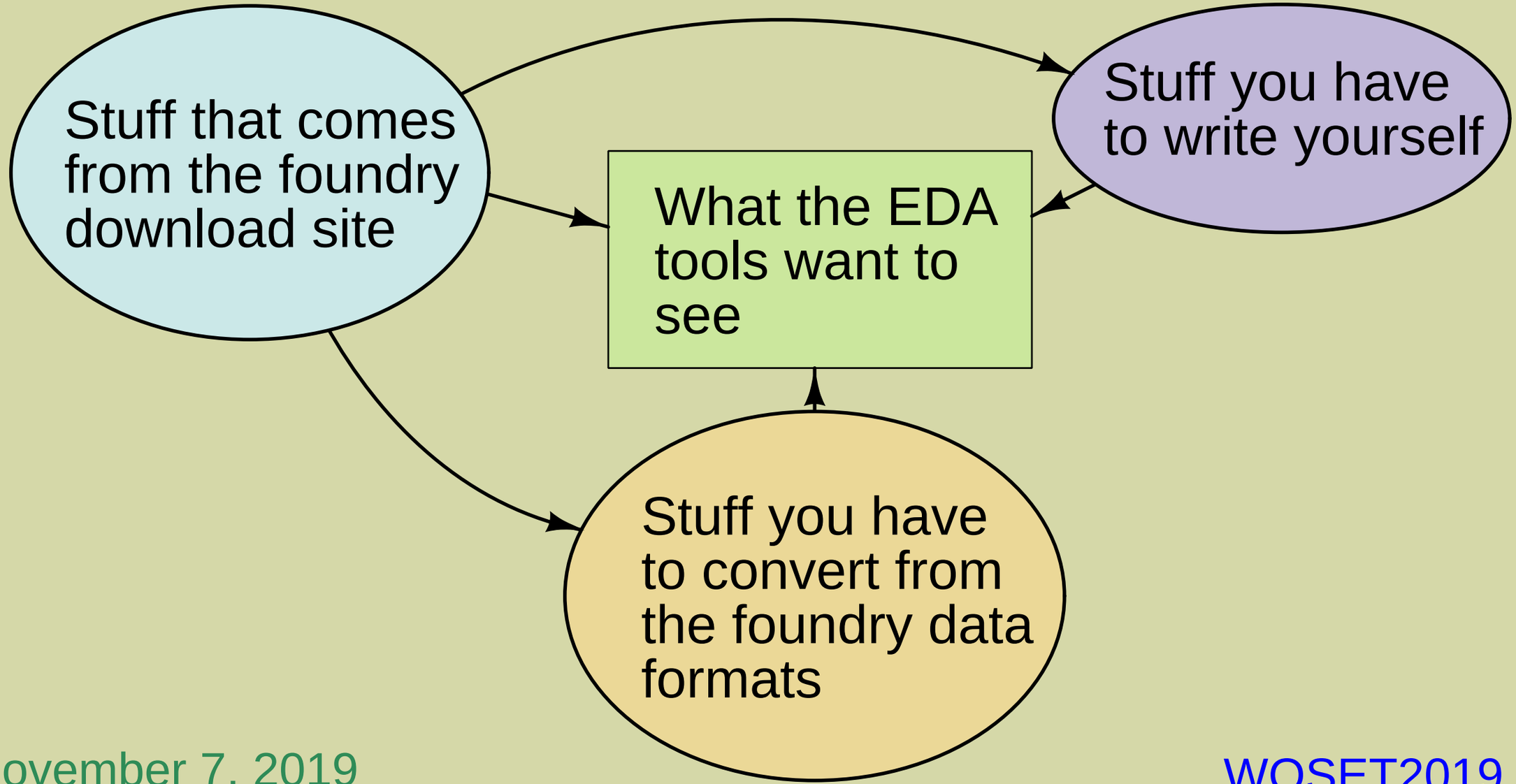
A Proposal for an Open PDK Framework

Solutions?

- Open source EDA tools with correct-by-design layout output
  (OpenROAD, OpenRAM)
- Translation services between obfuscated and foundry rules
  (efabless)
- Foundries with open processes
  (stay tuned. . .)
- A common framework for community-inspired tools
  (the rest of this talk)

A Proposal for an Open PDK Framework

Auto-PDKs (provisional name)



A Proposal for an Open PDK Framework

Auto-PDKs

Foundry Data

Libraries

- Digital Standard Cells
- Primitive Devices
- I/O Cells
- Analog IP
- Memory IP

Standard Formats

- Liberty
- LEF
- GDS
- CDL/SPICE
- Verilog
- Documentation

Start with foundry download data.

Hopefully this comes in standard formats.

A Proposal for an Open PDK Framework

Auto-PDKs

Foundry Data

Libraries

- Digital Standard Cells
- Primitive Devices
- I/O Cells
- Analog IP
- Memory IP

Standard Formats

- Liberty
- LEF
- GDS
- CDL/SPICE
- Verilog
- Documentation

Custom Data

Makefile

- FEOL/BEOL Options

Templates

- Magic techfile (.tech)
- Magic PDK (.tcl)
- Magic startup script (.magicrc)
- Netgen setup file (.tcl)
- Qflow setup file (.sh)
- Graywolf setup file (.par)

Makefile-driven system

Each open-source EDA tool will need to contribute template files for the process.

A Proposal for an Open PDK Framework

Auto-PDKs

Foundry Data

Libraries

Digital Standard Cells

Primitive Devices

I/O Cells

Analog IP

Memory IP

Standard Formats

Liberty

LEF

GDS

CDL/SPICE

Verilog

Documentation

Custom Data

Makefile

FEOL/BEOL Options

Templates

Magic techfile (.tech)

Magic PDK (.tcl)

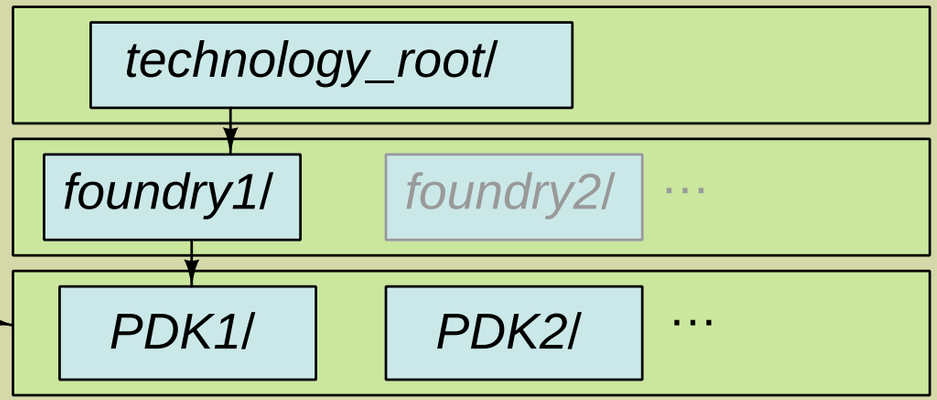
Magic startup script (.magicrc)

Netgen setup file (.tcl)

Qflow setup file (.sh)

Graywolf setup file (.par)

Installed PDKs



Different process option selections generate different PDKs

A Proposal for an Open PDK Framework

Auto-PDKs

Foundry Data

Libraries

- Digital Standard Cells
- Primitive Devices
- I/O Cells
- Analog IP
- Memory IP

Standard Formats

- Liberty
- LEF
- GDS
- CDL/SPICE
- Verilog
- Documentation

Custom Data

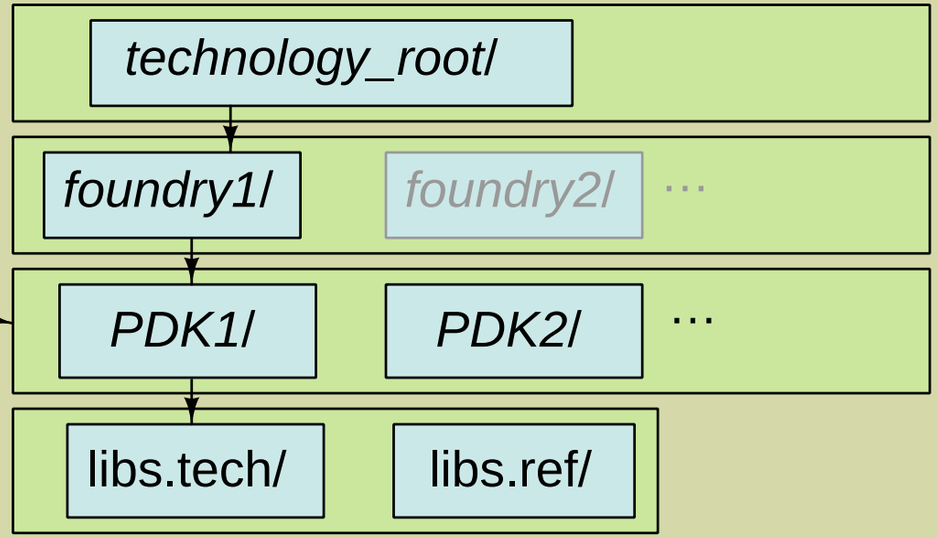
Makefile

- FEOL/BEOL Options

Templates

- Magic techfile (.tech)
- Magic PDK (.tcl)
- Magic startup script (.magicrc)
- Netgen setup file (.tcl)
- Qflow setup file (.sh)
- Graywolf setup file (.par)

Installed PDKs



Data are separated into technology information and IP libraries

A Proposal for an Open PDK Framework

Auto-PDKs

Foundry Data

Libraries

Digital Standard Cells

Primitive Devices

I/O Cells

Analog IP

Memory IP

Standard Formats

Liberty

LEF

GDS

CDL/SPICE

Verilog

Documentation

Custom Data

Makefile

FEOL/BEOL Options

Templates

Magic techfile (.tech)

Magic PDK (.tcl)

Magic startup script (.magicrc)

Netgen setup file (.tcl)

Qflow setup file (.sh)

Graywolf setup file (.par)

Installed PDKs

technology_root/

foundry1/ *foundry2/* …

PDK1/ *PDK2/* …

libs.tech/ libs.ref/

magic/
netgen/
qflow/

lef/

Technology information is per-tool when custom formats are required, or use standard formats.

A Proposal for an Open PDK Framework

Auto-PDKs

Foundry Data

Libraries

Digital Standard Cells

Primitive Devices

I/O Cells

Analog IP

Memory IP

Standard Formats

Liberty

LEF

GDS

CDL/SPICE

Verilog

Documentation

Custom Data

Makefile

FEOL/BEOL Options

Templates

Magic techfile (.tech)

Magic PDK (.tcl)

Magic startup script (.magicrc)

Netgen setup file (.tcl)

Qflow setup file (.sh)

Graywolf setup file (.par)

Installed PDKs

technology_root/

foundry1/ *foundry2/* …

PDK1/ *PDK2/* …

libs.tech/ libs.ref/

magic/
netgen/
qflow/

lef/

mag/
maglef/

lef/
liberty/
verilog/
gds/
spi/

Libraries have custom or standard formats as needed.

A Proposal for an Open PDK Framework

Benefits of an open framework

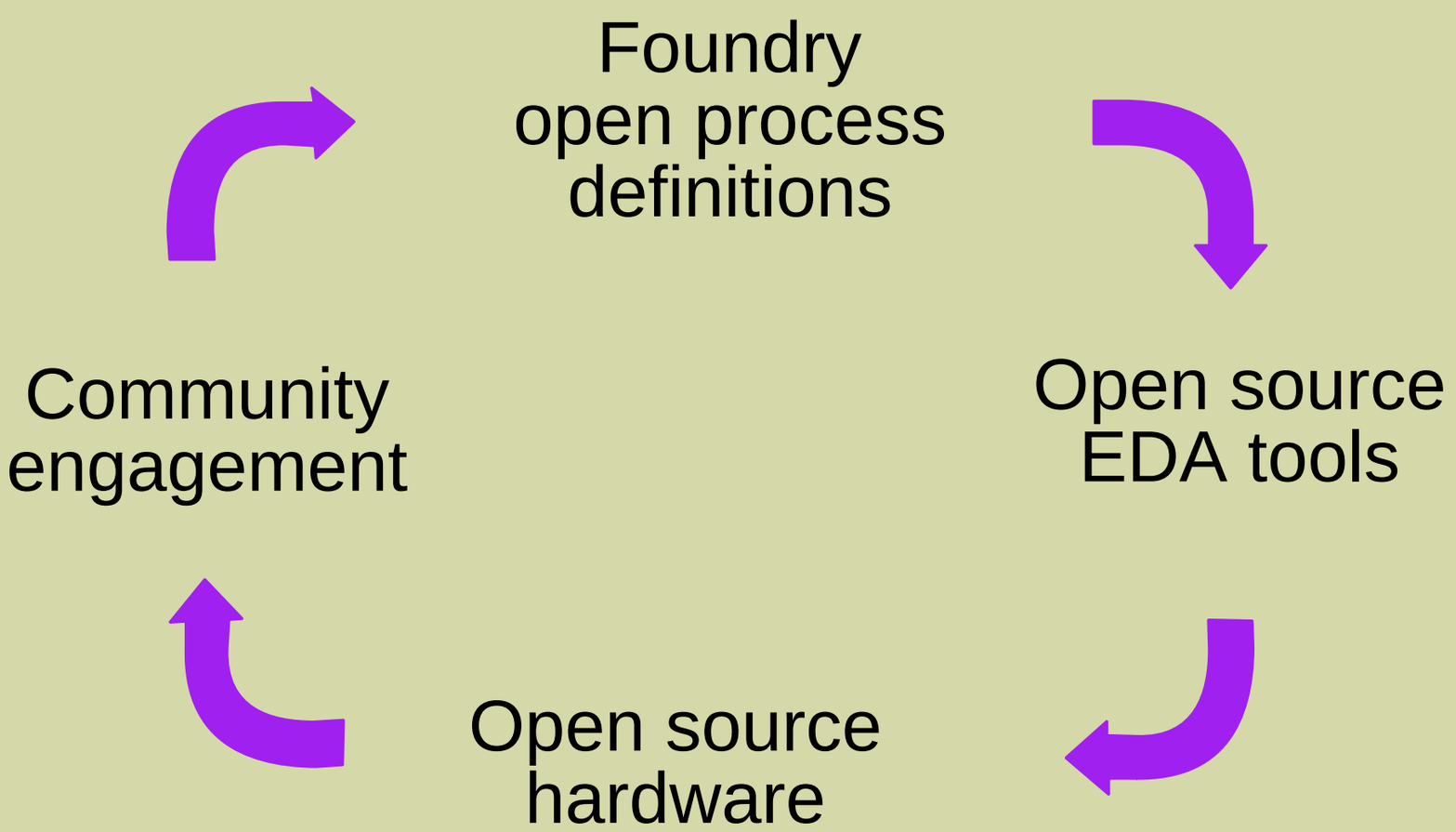
- Easier development of open source EDA tools
- Set of example (real) processes always available
- For open PDKs, direct download from repo like github
- More community development of tools and PDKs
- More community use of open source EDA tools
- More community design, more creativity

The New Golden Age of Open Silicon

- A brief history of SCMOS
- A proposal for an open PDK Framework
- The New Golden Age of Open Silicon

The New Golden Age of Open Silicon

The Virtuous Cycle



The New Golden Age of Open Silicon

Clever design can beat raw performance.

Open community design = massively distributed design

Massively distributed design is more valuable than any proprietary algorithm in a CAD tool.

Open source EDA tools, open process PDKs, and open hardware will create the New Golden Age of Open Silicon.

The New Golden Age of Open Silicon

November 7, 2019

WOSET2019 ICCAD