2023
DESIGN AND VERIFICATION™
DVCON
CONFERENCE AND EXHIBITION
EUROPE
10 YEAR ANNIVERSARY

The Three Body Problem

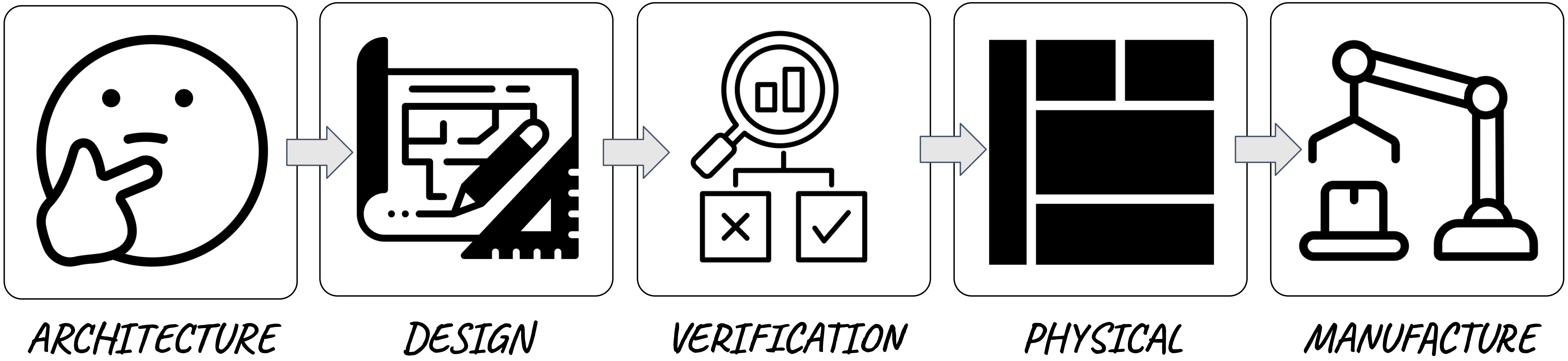
There’s more to building Silicon than EDA currently helps

Peter Birch (Vypercore) & Ben Marshall (PQShield)

vypercore PQ SHIELD

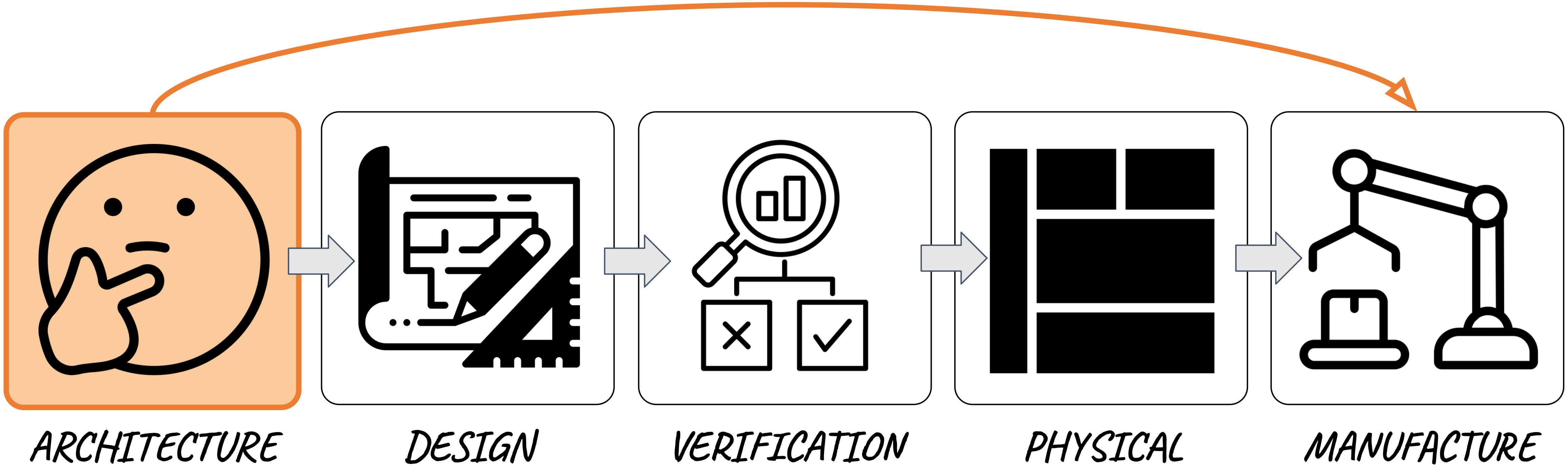
accellera
SYSTEMS INITIATIVE

What’s the problem?

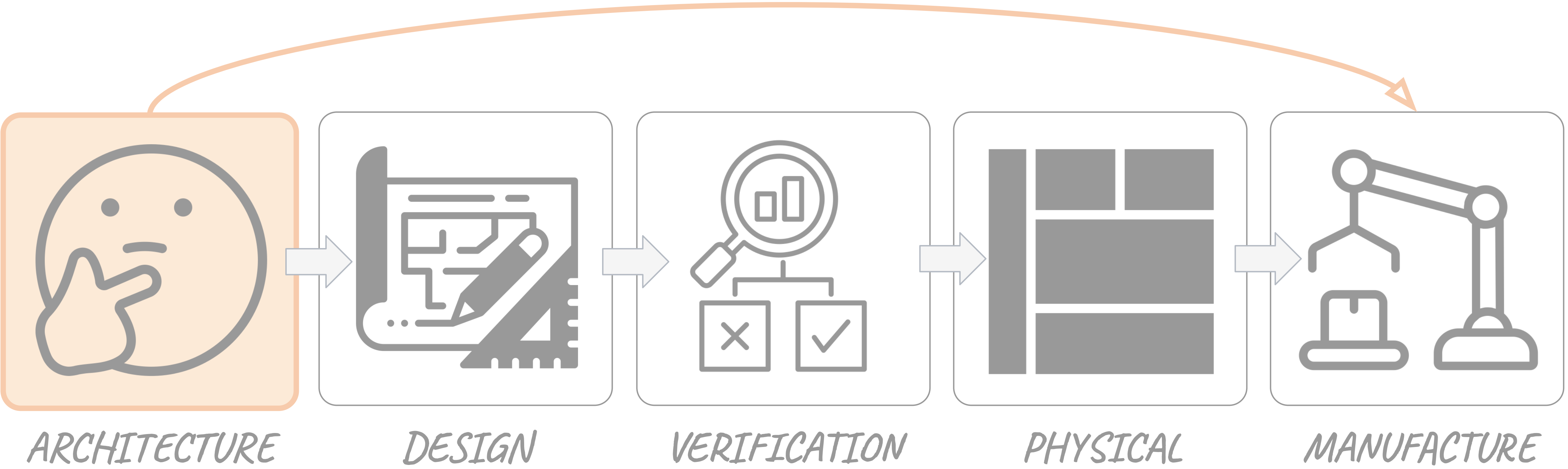


ARCHITECTURE → DESIGN → VERIFICATION → PHYSICAL → MANUFACTURE

What’s the problem?



What's the problem?



PROVENANCE

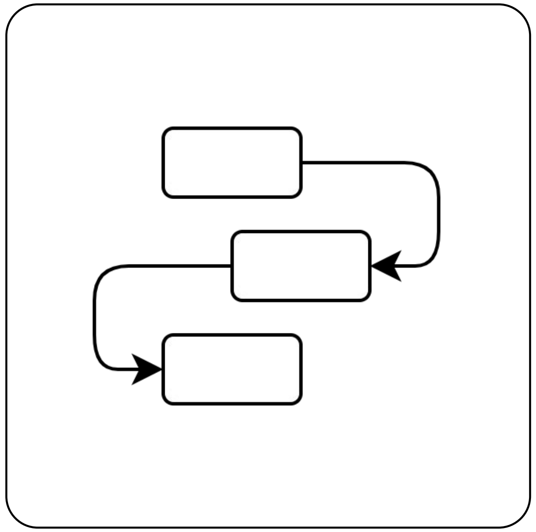
The Three Bodies

- The ‘big three’ EDA vendors don’t make this easy
- Tools lack common interfaces
- Proprietary inputs and outputs
- Results vary run-to-run
- NDAs stop flows being shared

So what do we do?

SUCCESSFUL

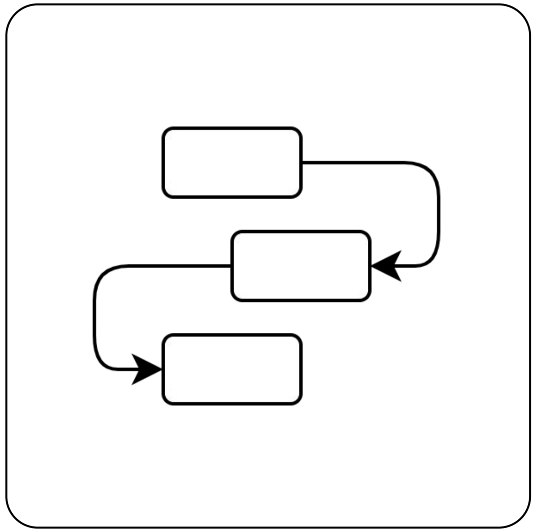
Tenets of a ^Silicon Engineering Flow



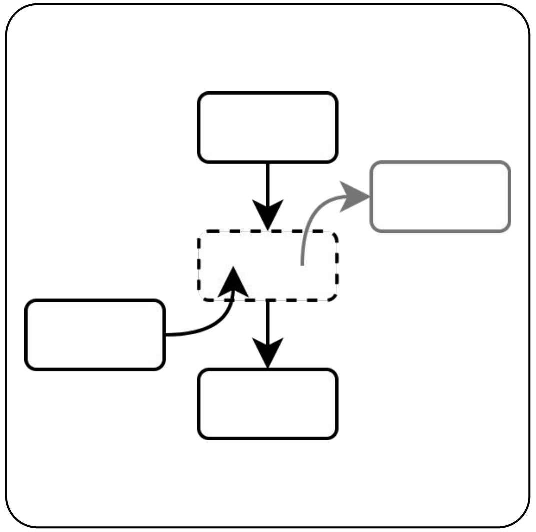
TRACEABLE

SUCCESSFUL

Tenets of a ^Silicon Engineering Flow

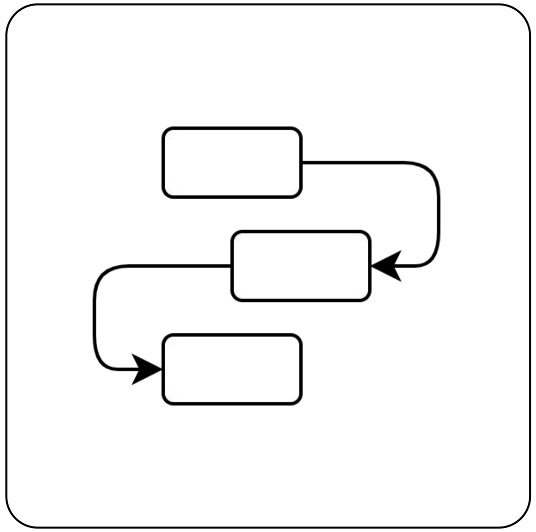


TRACEABLE

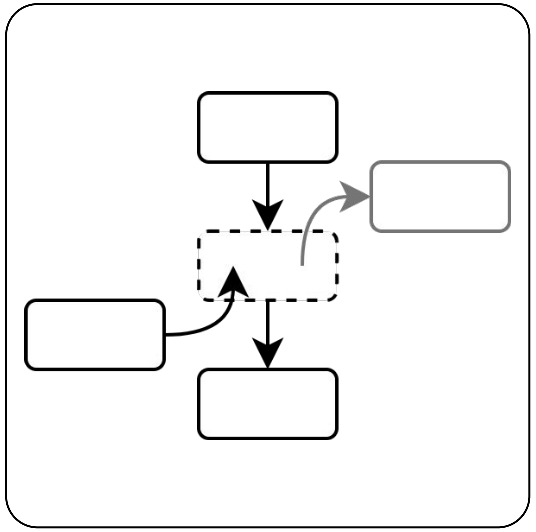


MODULAR

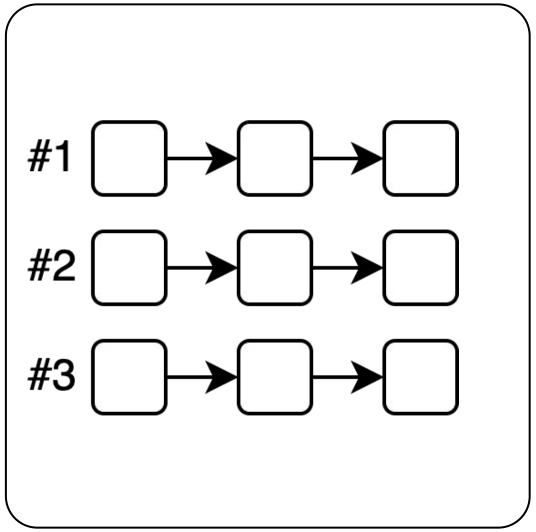
Tenets of a SUCCESSFUL Silicon Engineering Flow



TRACEABLE

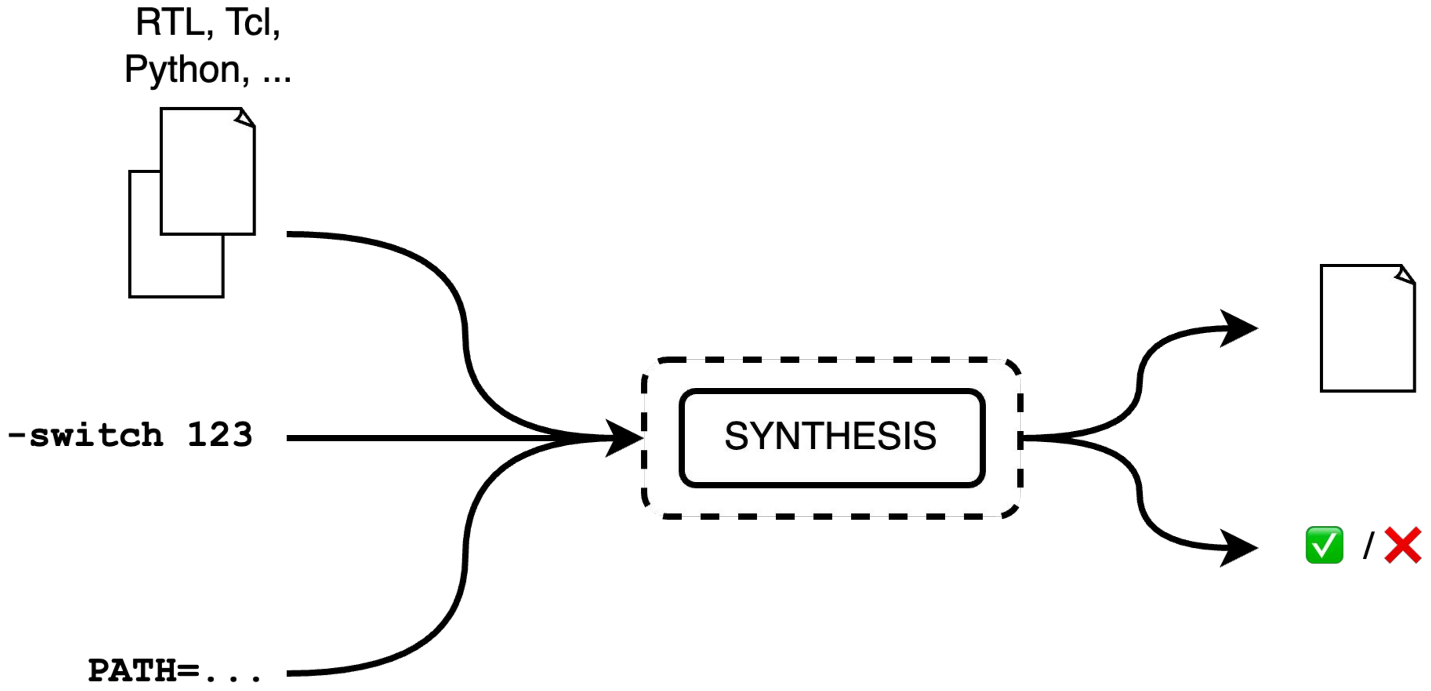


MODULAR



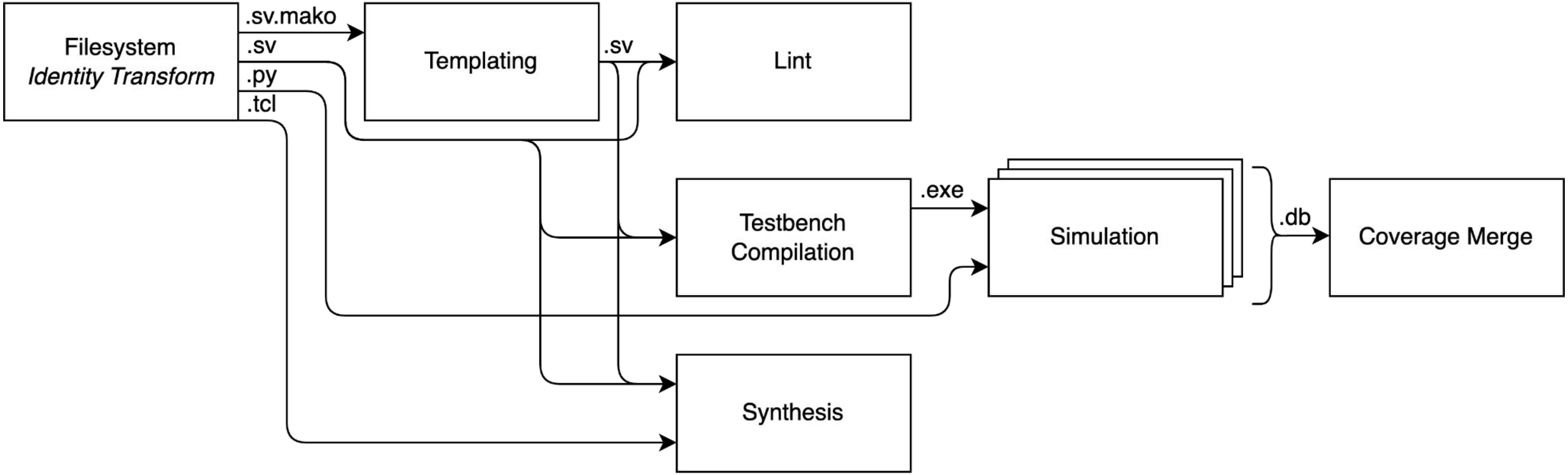
REPRODUCIBLE

Pure Functions

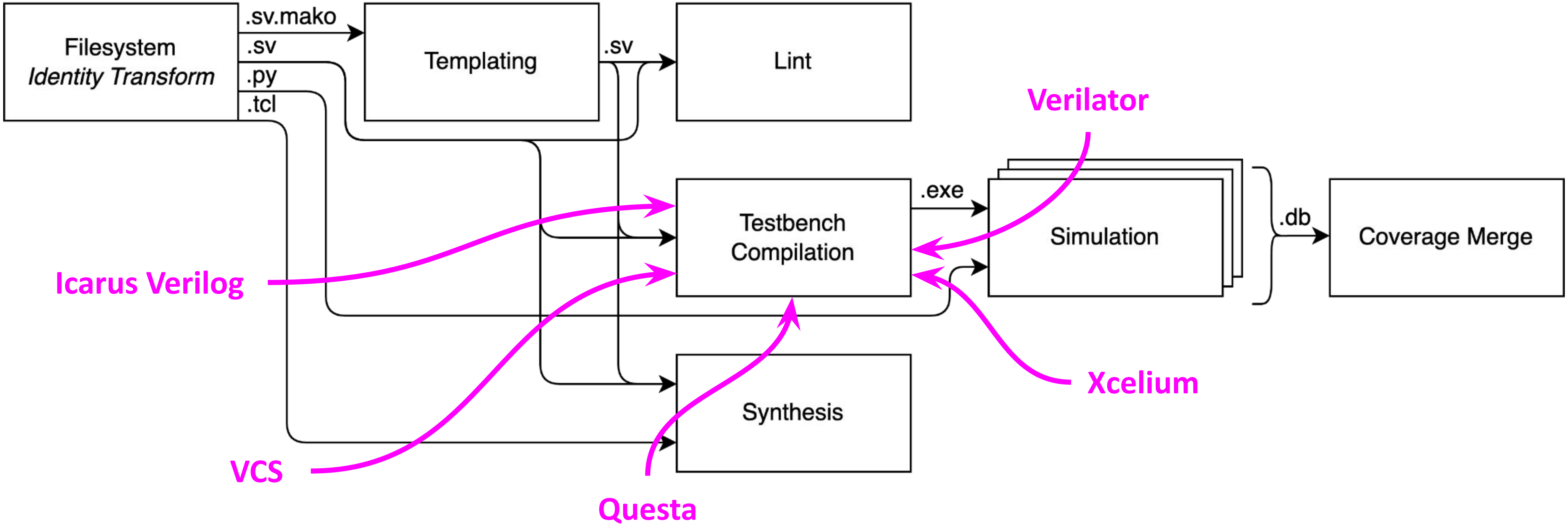


```
(files, pass) = synthesis(files, switches, environment)
```

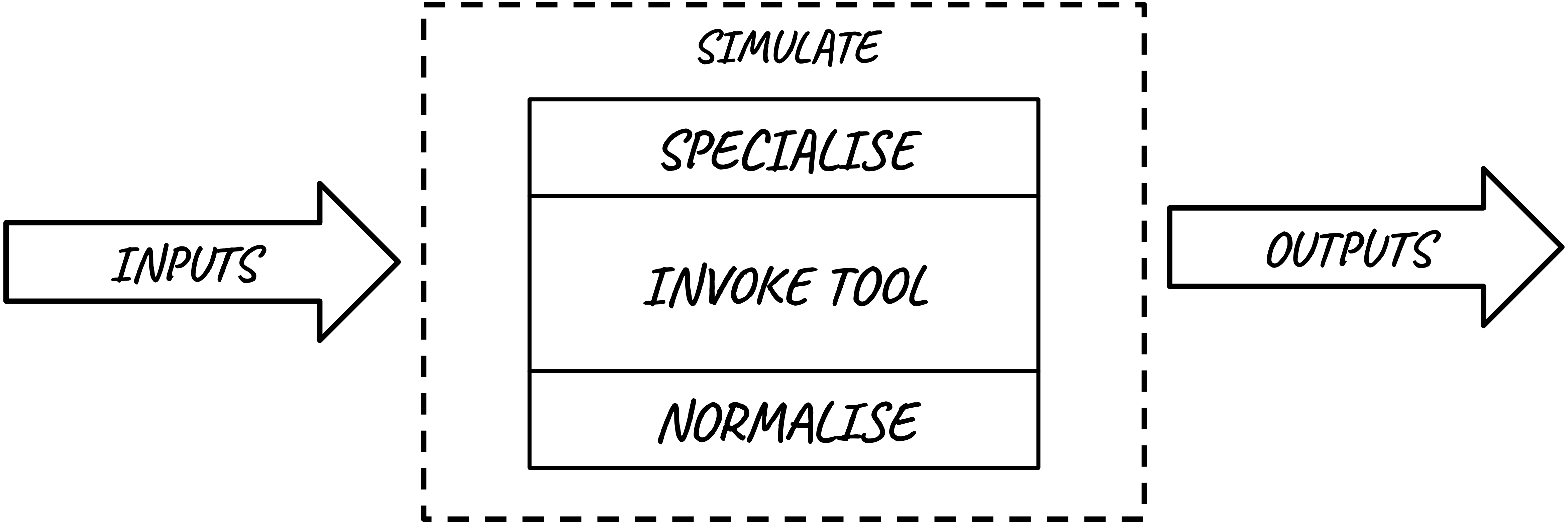
Flows are Graphs



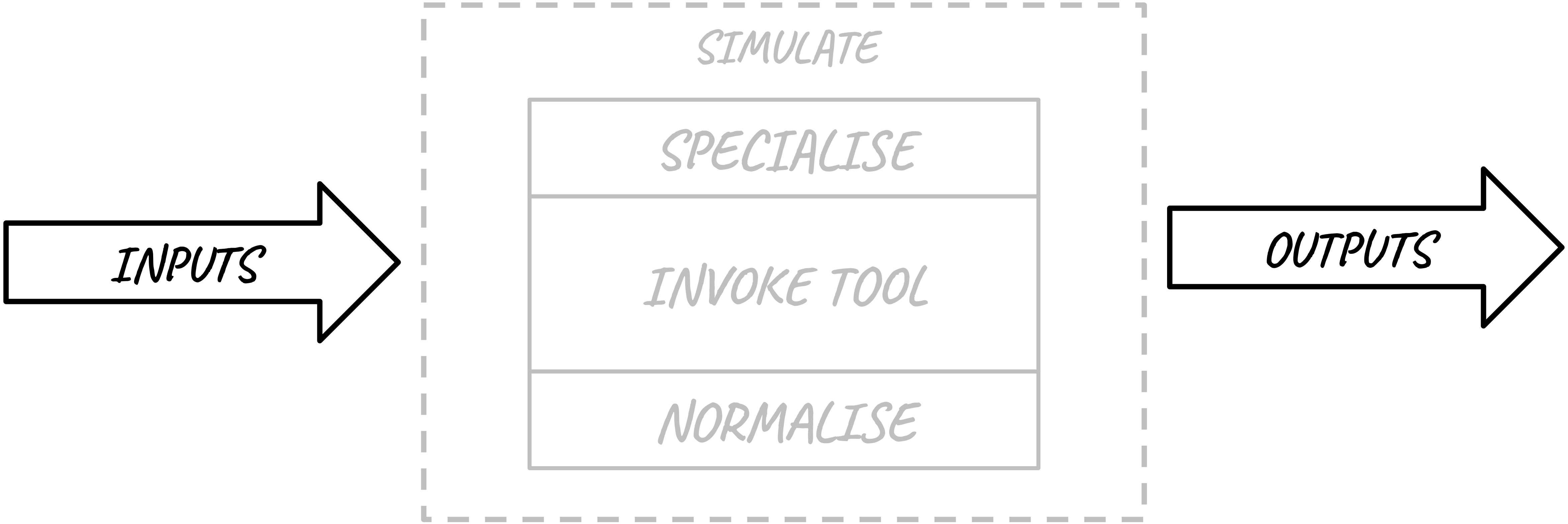
Nodes


Filesystem
Identity Transform
.sv.mako
.sv
.py
.tcl
Templating
.sv
Lint
Verilator
Testbench Compilation
.exe
Simulation
.db
Coverage Merge
Icarus Verilog
Xcelium
Synthesis
VCS
Questa

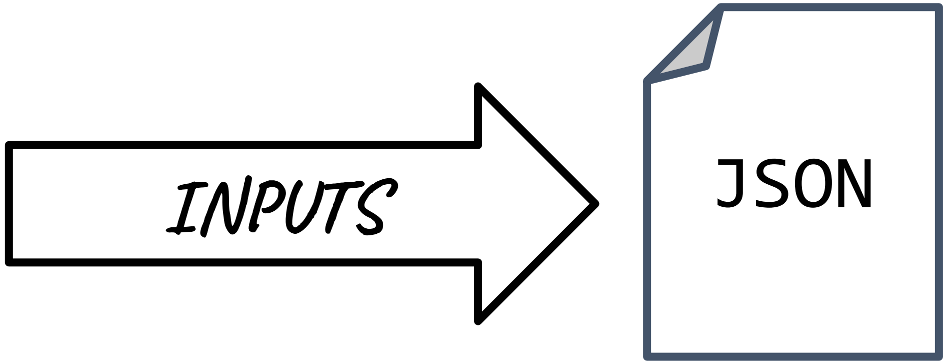
Transformations



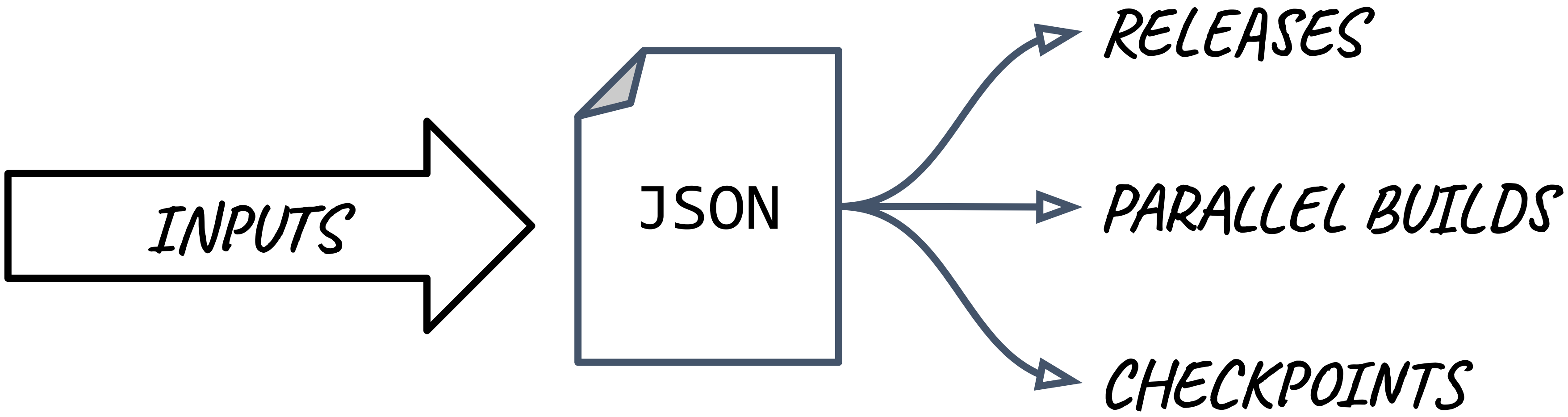
Interfaces


INPUTS
SIMULATE
SPECIALISE
INVOKE TOOL
NORMALISE
OUTPUTS

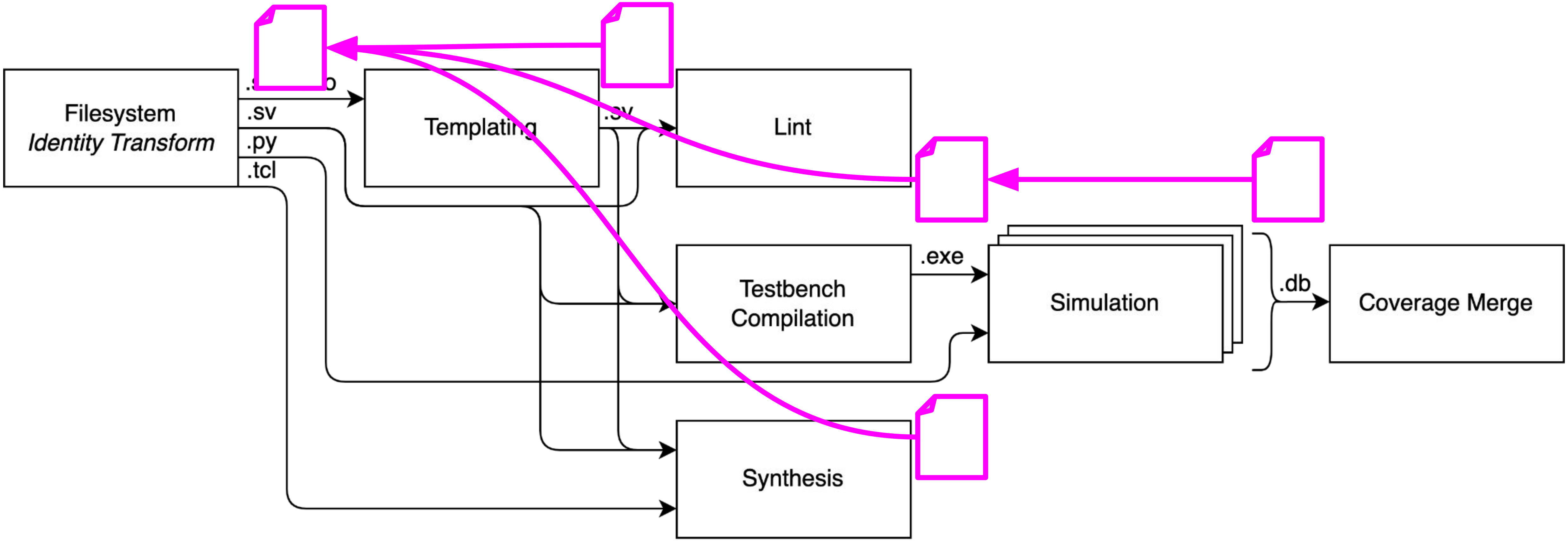
Interfaces



Interfaces



Tracking References



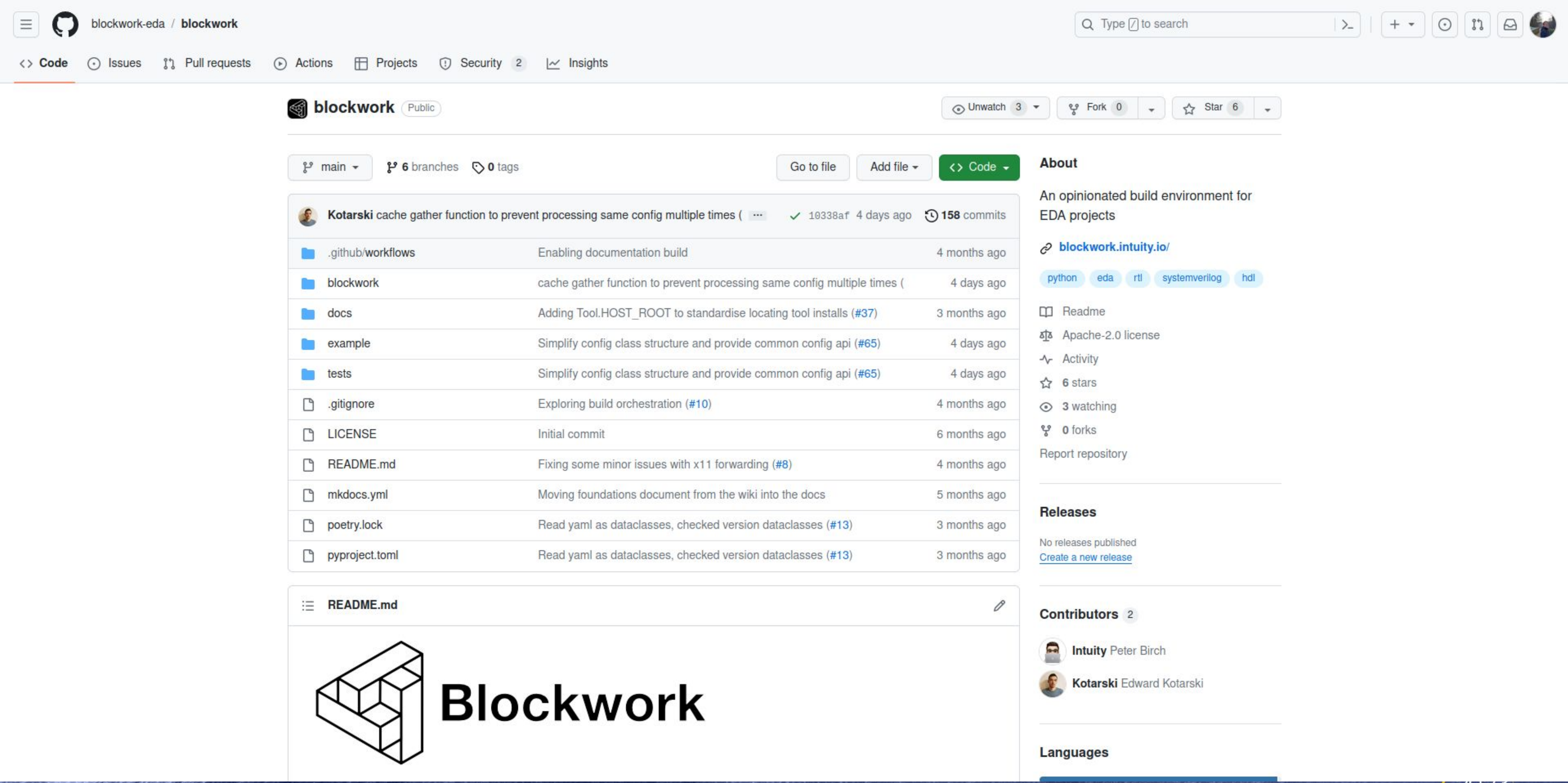
Makefiles are not enough

- EDA flows are very complex
- Many inputs and outputs
- Deeply branched trees of transformations
- Software build tools are not well suited

So we need something better…

…we have an answer…

…and it’s open source 🤯



Kotarski cache gather function to prevent processing same config multiple times (··· ✓ 10338af 4 days ago 158 commits

.github/workflows	Enabling documentation build	4 months ago
blockwork	cache gather function to prevent processing same config multiple times (	4 days ago
docs	Adding Tool.HOST_ROOT to standardise locating tool installs (#37)	3 months ago
example	Simplify config class structure and provide common config api (#65)	4 days ago
tests	Simplify config class structure and provide common config api (#65)	4 days ago
.gitignore	Exploring build orchestration (#10)	4 months ago
LICENSE	Initial commit	6 months ago
README.md	Fixing some minor issues with x11 forwarding (#8)	4 months ago
mkdocs.yml	Moving foundations document from the wiki into the docs	5 months ago
poetry.lock	Read yaml as dataclasses, checked version dataclasses (#13)	3 months ago
pyproject.toml	Read yaml as dataclasses, checked version dataclasses (#13)	3 months ago

About

An opinionated build environment for EDA projects

blockwork.intuity.io/

python eda rtl systemverilog hdl

Readme
Apache-2.0 license
Activity
6 stars
3 watching
0 forks
Report repository

Releases

No releases published
Create a new release

Contributors 2

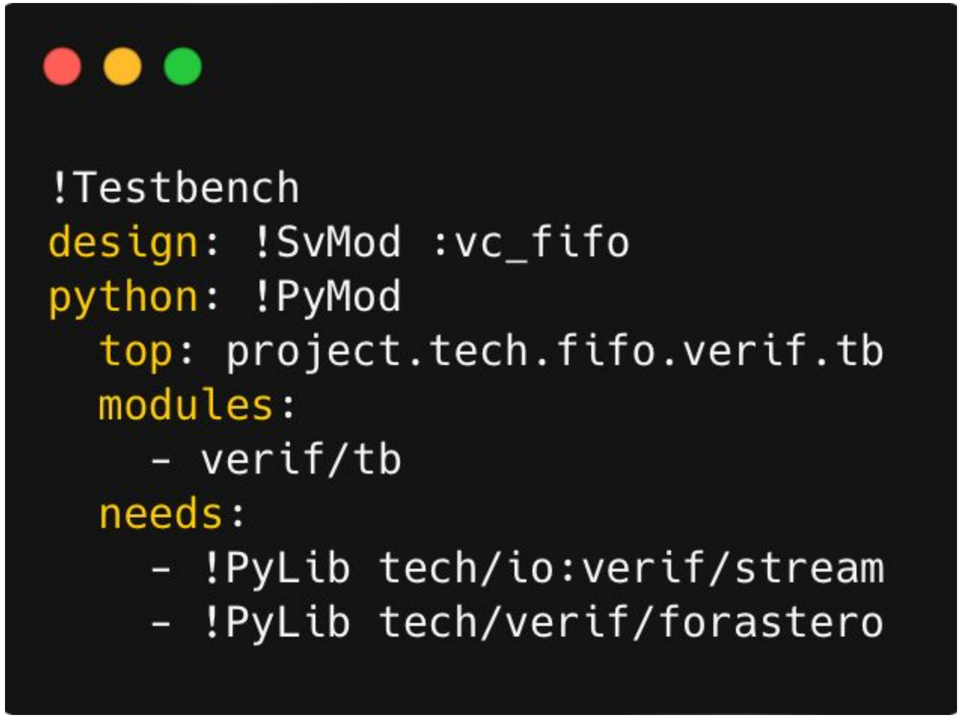
Intuity Peter Birch
Kotarski Edward Kotarski

Languages

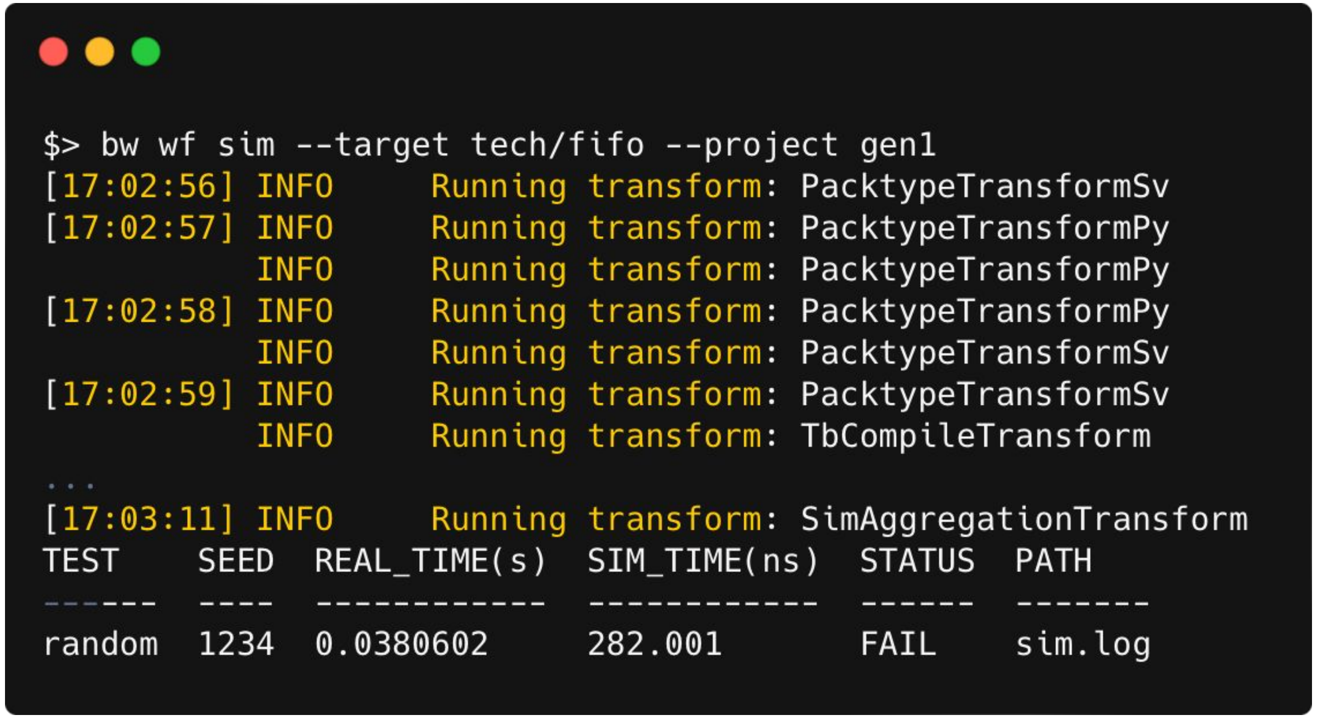
Blockwork

- Tool-agnostic framework
- Written in Python
- Uses Docker to isolate each step
- You bring the tool wrappers, transforms, and flows
- Under *active* and *rapid* development by VyperCore
- Inspired by Edalize and FuseSoC

Blockwork



```
!Testbench
design: !SvMod :vc_fifo
python: !PyMod
  top: project.tech.fifo.verif.tb
  modules:
    - verif/tb
  needs:
    - !PyLib tech/io:verif/stream
    - !PyLib tech/verif/forastero
```

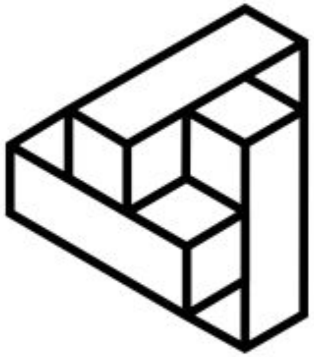


```
$> bw wf sim --target tech/fifo --project gen1
[17:02:56] INFO     Running transform: PacktypeTransformSv
[17:02:57] INFO     Running transform: PacktypeTransformPy
           INFO     Running transform: PacktypeTransformPy
[17:02:58] INFO     Running transform: PacktypeTransformPy
           INFO     Running transform: PacktypeTransformSv
[17:02:59] INFO     Running transform: PacktypeTransformSv
           INFO     Running transform: TbCompileTransform
...
[17:03:11] INFO     Running transform: SimAggregationTransform
TEST    SEED  REAL_TIME(s)  SIM_TIME(ns)  STATUS  PATH
------  ----  ------------  ------------  ------  -------
random  1234  0.0380602     282.001       FAIL    sim.log
```

Runs here, there, or anywhere…

DEBUG @ 25,000 FT

Blockwork



Please come join in the fun!

github.com/blockwork-eda

In Summary

- A good silicon flow is traceable, modular, and reproducible
- We need better tools!
- Blockwork is tool-agonistic and open source
- Vendors - please help us:
  - Relax NDAs around flows
  - Standardised interfaces
  - Machine readable inputs and outputs
  - Fully reproducible transformations