How far can FOSS take you?

ORConf 2025

Advocated for silicon startups to use & contribute to open source tools

Two years ago I helped put an open methodology at the heart of a startup's engineering process

The startup failed

The startup failed, so was I wrong?

Weighing it Up

Reveals our secrets

Gives others a leg up

Wastes time

Makes it easier to leave

Weighing it Up

Reveals our secrets but many problems aren't unique

Gives others a leg up so others can help you

Wastes time but docs and unit tests help you too

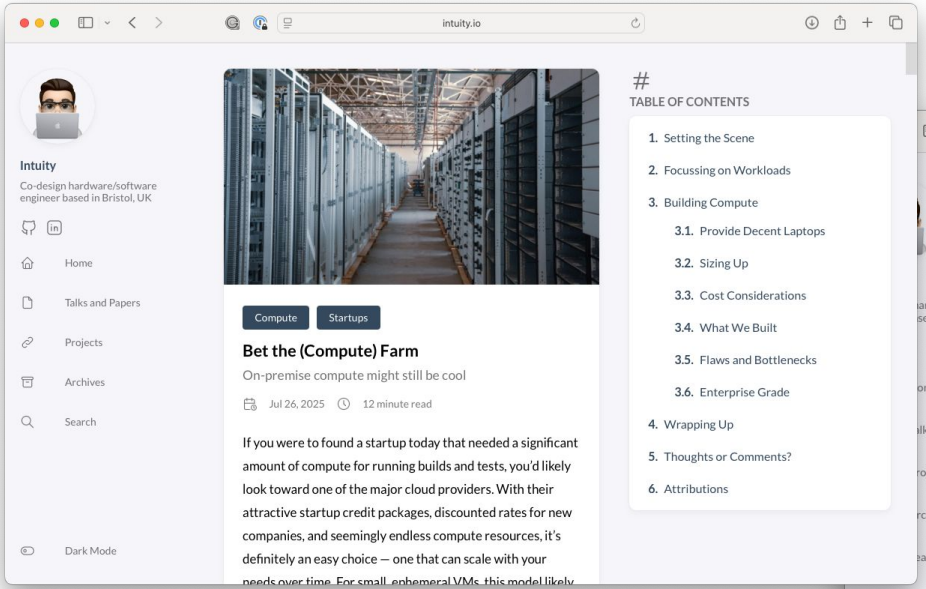
Makes it easier to leave or motivates you to stay

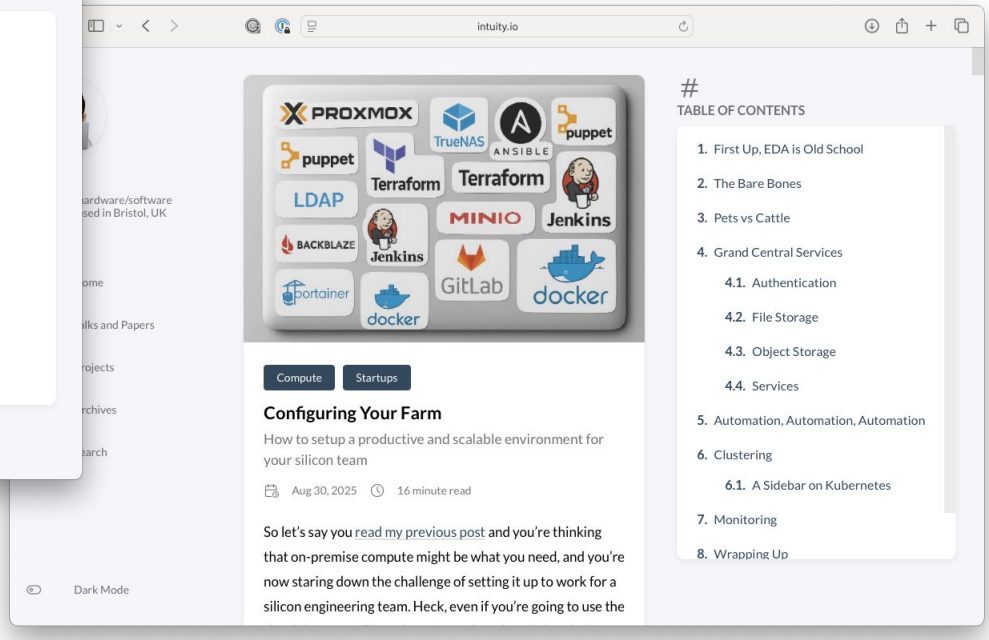
So what did I learn?

1. Infrastructure and Compute

- Complex projects require careful setup
- Understanding that infrastructure is equally flow and compute
- Spend wisely, cloud costs ramp up *fast*
- Use infrastructure-as-code (Ansible/Puppet/Terraform)
- The best computer is the one you have with you

1. Infrastructure and Compute





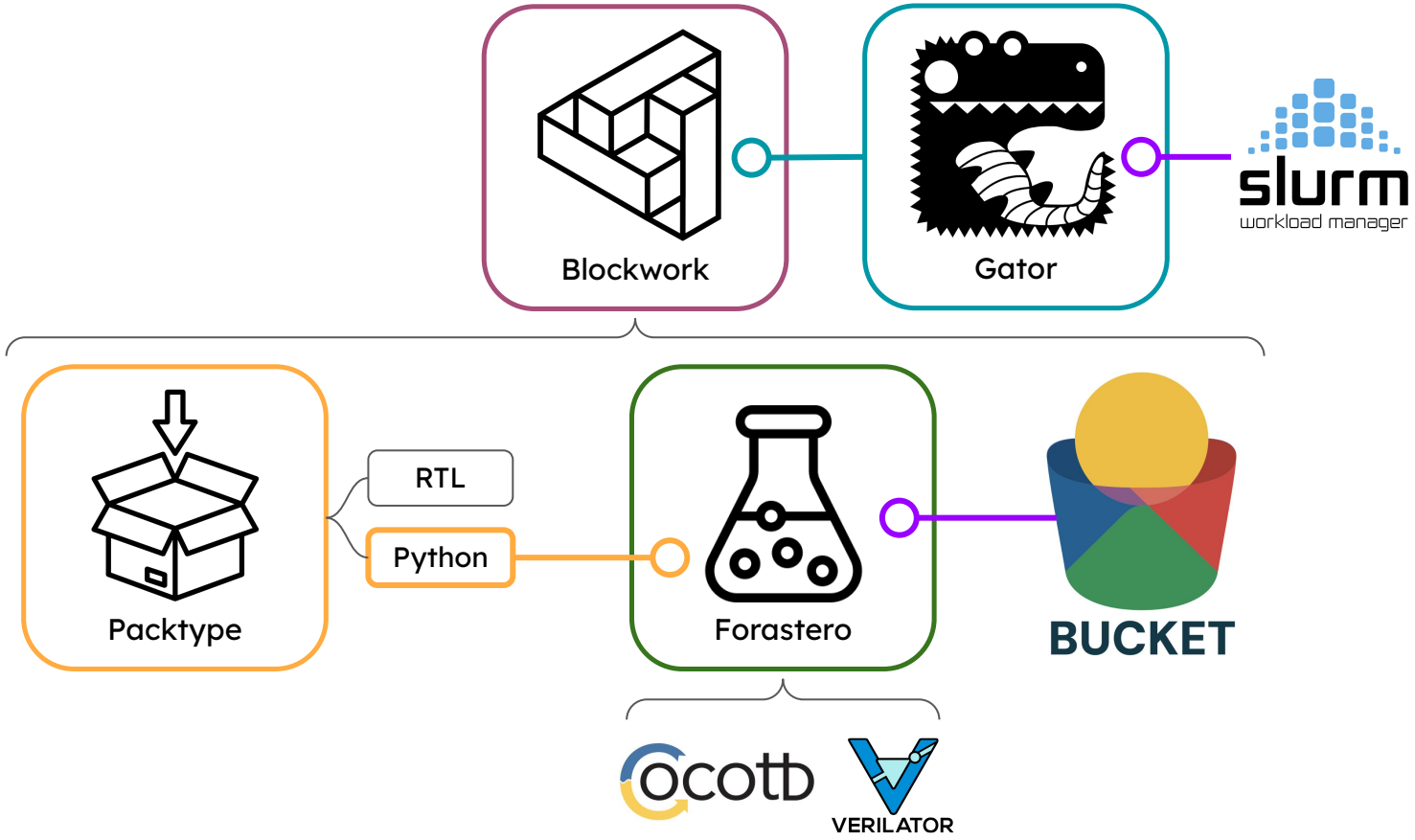
intuity.io

2. Keep Tools Focussed

Photo by Denise Jans on Unsplash

2. Keep Tools Focussed

Photo by Denise Jans on Unsplash


Blockwork
Gator
slurm
workload manager
Packtype
RTL
Python
Forastero
BUCKET
ocotb
VERILATOR

3. The Flow is Incomplete

- Icarus, Verilator, cocotb, GTKWave, Surfer, Yosys, Sby, … are amazing
- What to use for CDC, RDC, DFT (BIST), …?
- Make your flows modular using Edalize, Bazel, …
- Fix rough edges and share them

Wrapping Up

- Open tools and methodologies will get you *a long way*
- What now? FRACTILE
- Building AI inference accelerators using open tools and flows
  - *apytypes, bazel, bucket, cocotb, constrainedrandom, gator, gtkwave, icarus, packtype, slang, surfer, verilator, yosys*

intuity.io

fractile.ai