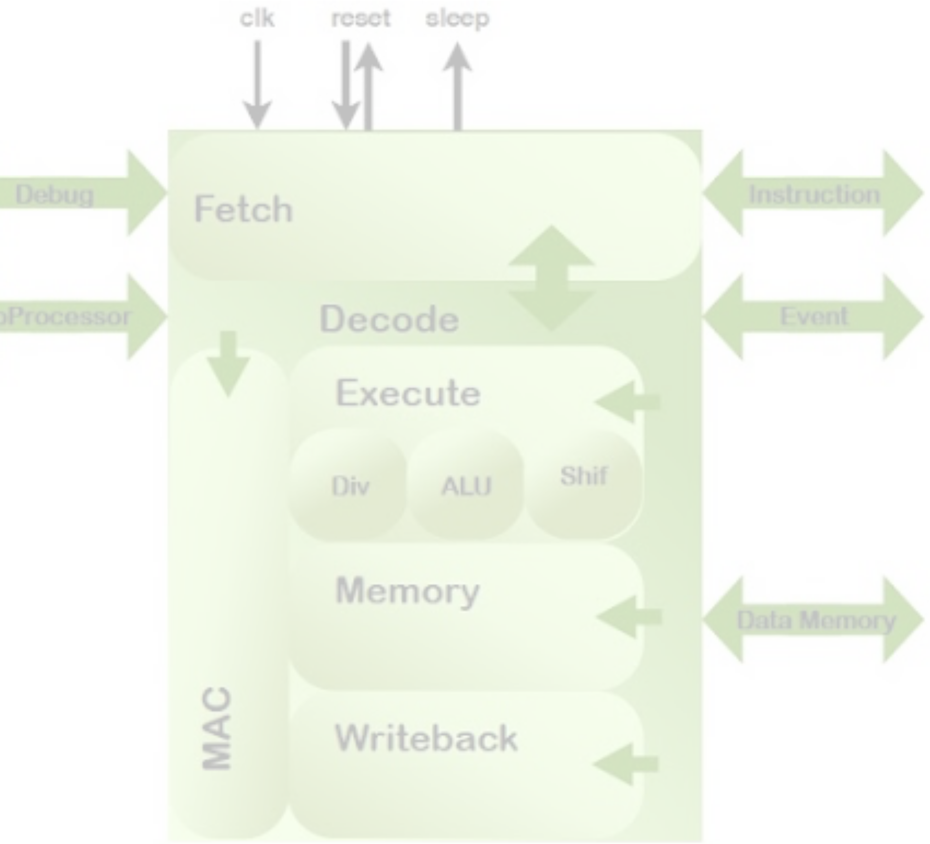
CORE SEMICONDUCTOR

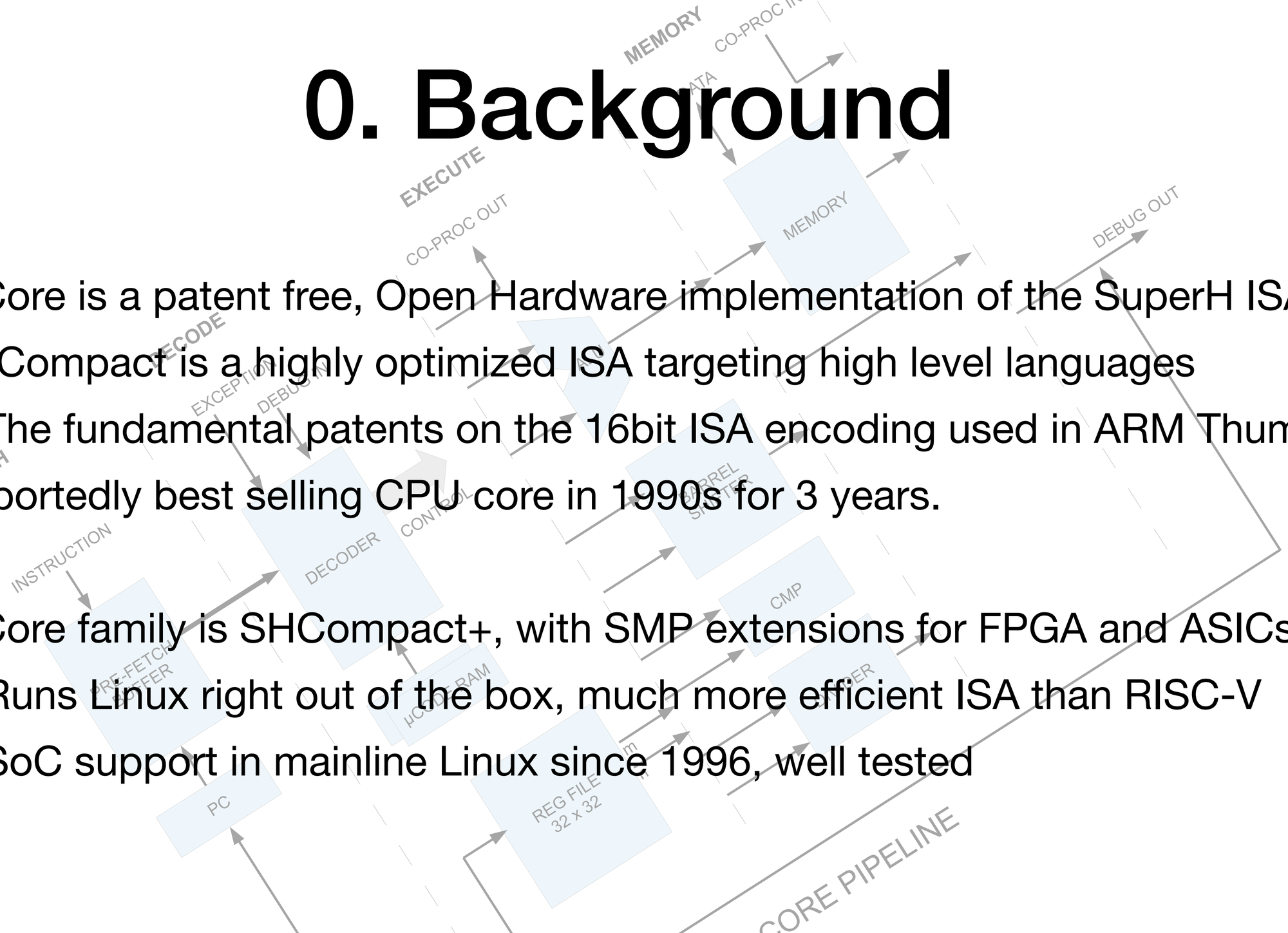
Preparing to tape out a fully open J-Core SoC on fully open tools



Rob W. Landley, D. Jeff Dionne
Feb 2022

J-core

0. Background



- J-Core is a patent free, Open Hardware implementation of the SuperH ISA
- SHCompact is a highly optimized ISA targeting high level languages
  - The fundamental patents on the 16bit ISA encoding used in ARM Thumb
- Reportedly best selling CPU core in 1990s for 3 years.

- J-Core family is SHCompact+, with SMP extensions for FPGA and ASICs
  - Runs Linux right out of the box, much more efficient ISA than RISC-V
  - SoC support in mainline Linux since 1996, well tested

Roadmap: What comes out from this talk?

- Upward compatible processor family:
  - ➡ J0: a 16 bit datapath, 15k ASIC gates (target). 3 stage pipeline
  - ✓ J1: Simple but full 32bit machine. 30k ASIC gates. 5 stage pipeline
    - ✓ J1sec: Trusted Compute Engine
  - ✓ J2: Full Harvard, cache, SMP capable. 50k gates. Parallel MAC
    - ✓ J2SMP: Linux capable Symmetric Multi-Processor
  - ✓ J32: Virtual Memory (MMU). 100k ASIC gates
  - ➡ J32FM: Multi-issue, FPU. 200k ASIC gates (target).

✓ SoC Platform, Peripheral Library, SoC interconnect and top-level generators

Why did we build it?

- J-Core is purpose built: for connected embedded systems
  - Not an academic exercise: Deployed in ‘Critical Infrastructure’, electrical grids

- Objective: To democratize the space: When knowledge and tools are available…
  - People can achieve their own goals
  - Leads to Info Security, Continuity of Business, and Advancement of the Art.

- We built this so we could build products with it, and so can anyone else.
  - In the end, all the way through to affordable, accessible ASIC based designs

Simple Application Example: VPN

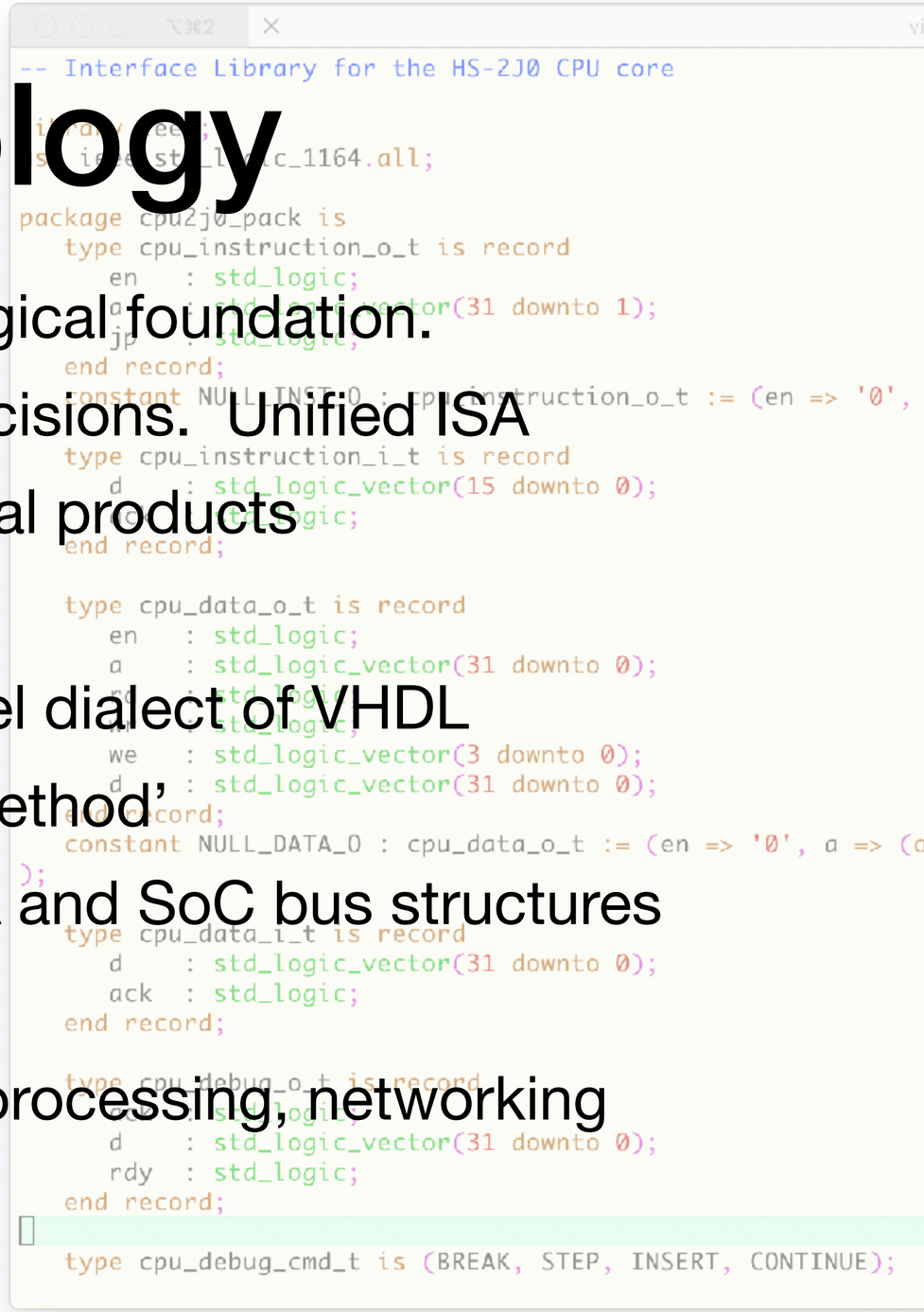
- Hardware crypto accelerated VPN device for remote telework (pandemic product)
  - J2SMP running Linux
  - Hardware cryptographic engine, coupled to a J1 trusted compute processor
  - On-SoC HSM functionality, and high speed.
- Prototype in FPGA took only a few weeks, new boards took a few months.
  - ASIC path when and only if the project reaches product maturity.

Good Technology

- SHCompact is an excellent and solid technological foundation.
  - Well researched and documented design decisions. Unified ISA
  - Multiple iterations: experience from use in real products

- J-Core is a clean reimplementation, in high level dialect of VHDL
  - Designed with the ESA/Gaisler ‘2 Process Method’
  - Extensive use of code generators for the ISA and SoC bus structures

- Implemented from the ground up to do signal processing, networking
  - Designed from the ground up to run Linux



Rationale: Why go to ASIC?

- Know your devices, own your data
- Security
- Enable innovation and development, our own and in the community

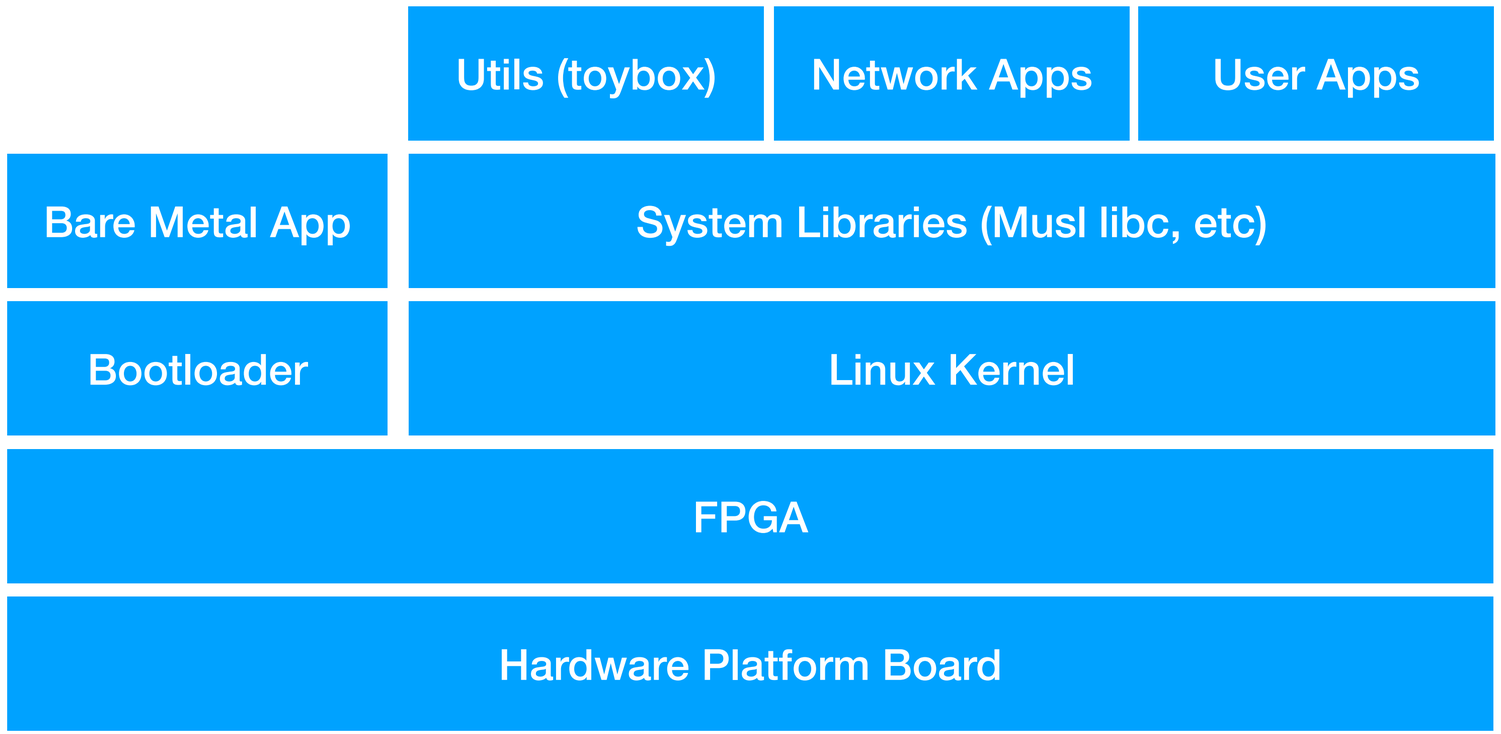
1. Open Source and Open Hardware

- Completely Open implementations and driving philosophy
  - J-Core RTL and tools are available under appropriate Open licenses
- Hardware platforms to run it are available under open licenses
  - More than just a reference design
- Complete, mature toolchains for ‘bare metal’ and Linux OS
- Boot code, Linux OS, Application examples…

HW Development Environment

- More than a development board, you can build real prototypes
  - Add modules to the COTS base board. Simple interface circuitry.
- Turtle Board represents a sort of ‘ideal’ J-Core Embedded device platform
  - 3rd party small boards like the UPDuino 2.0 useful also

RTL, HW and SW Complete



Feels like any Linux system.

Familiar set of tools and libraries

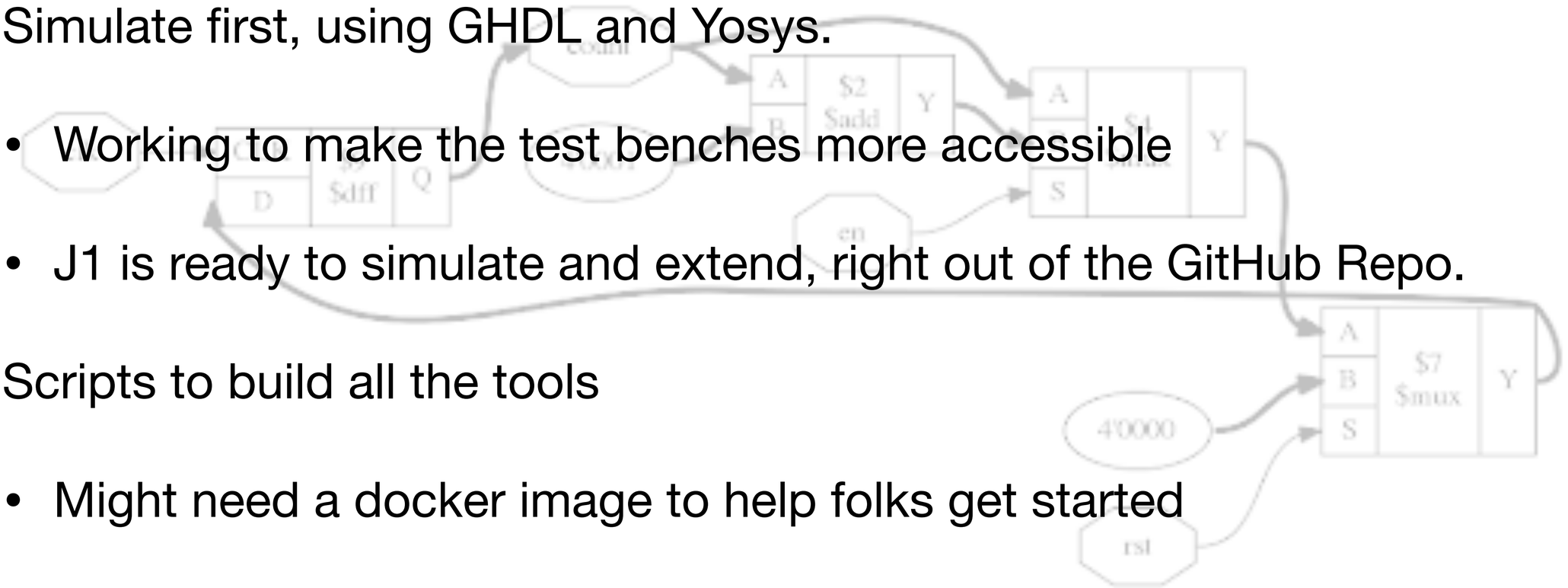
Boot code, Linux port, Device Drivers

Jx CPU, SoC generator, Peripherals

Hardware Designs in EAGLE CAD

2. Tools and Components

- Simulate first, using GHDL and Yosys.
  - Working to make the test benches more accessible
  - J1 is ready to simulate and extend, right out of the GitHub Repo.
- Scripts to build all the tools
  - Might need a docker image to help folks get started
- GHDL+Yosys+NextPNR go all the way to FPGA for Lattice



GHDL

2(3) Options for Tool Flow

DARPA

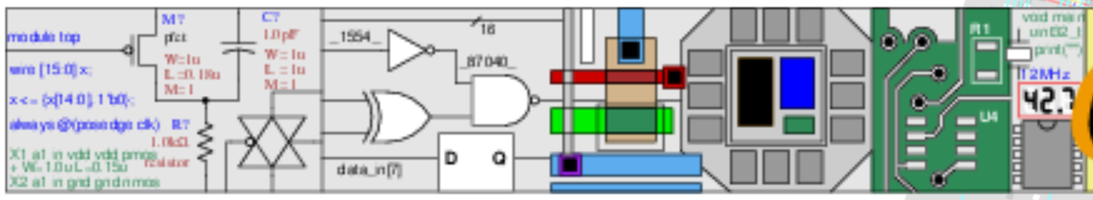
OpenROAD

Google

OpenLane

+ skywater TECHNOLOGY

FOSS 130nm Production PDK



open_pdks

Qflow 1.4

An Open-Source Digital Synthesis Flow

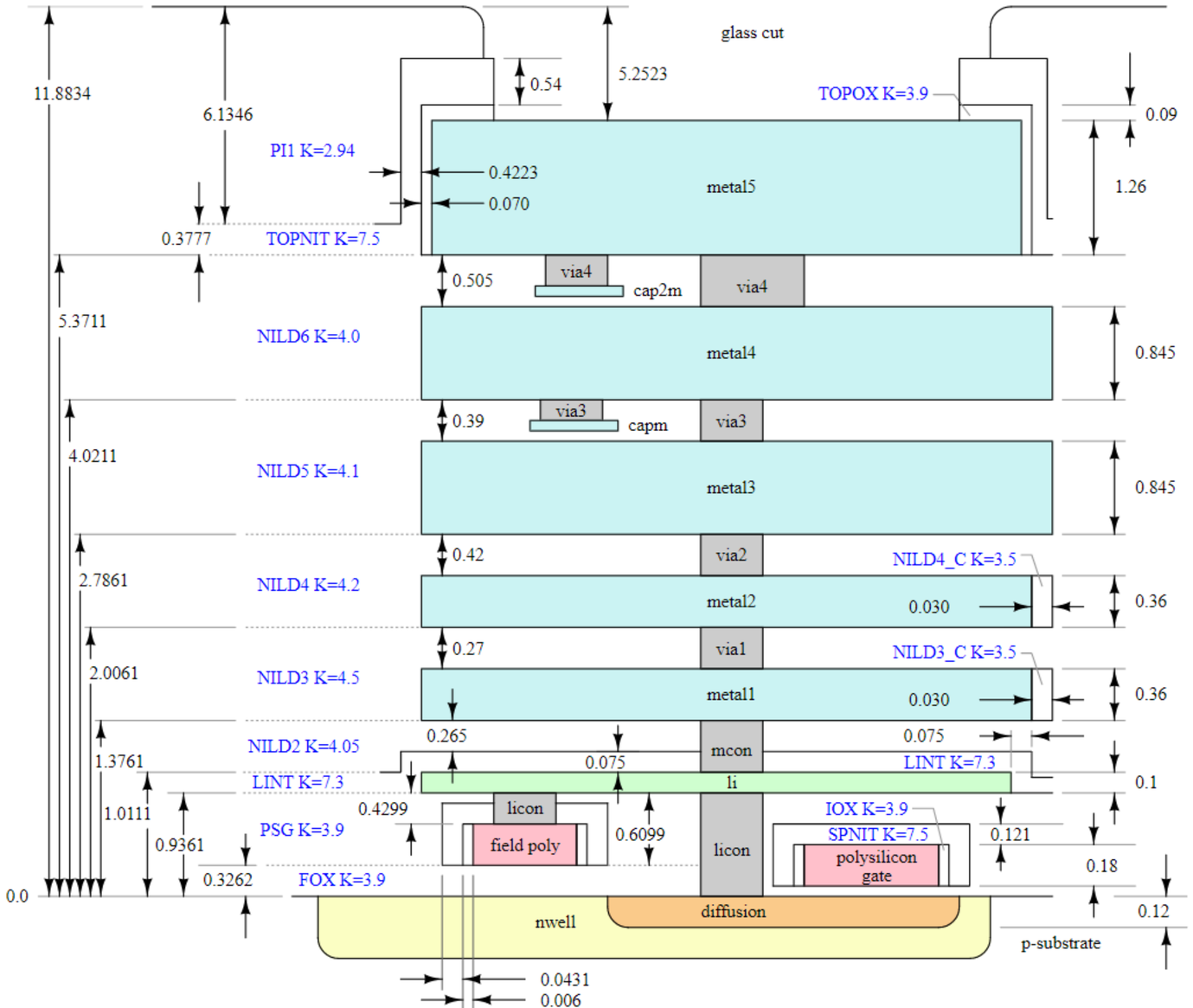
3. Fabrication Options

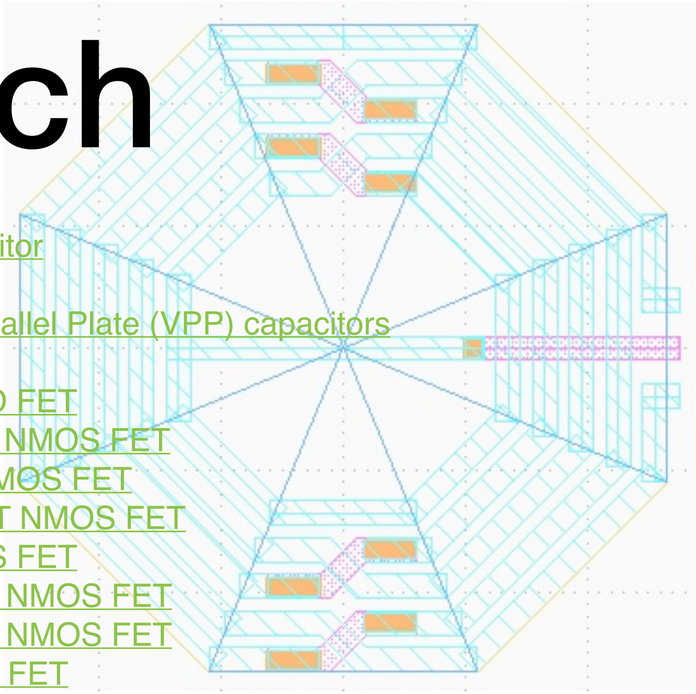
- Multiple routes:
  - Older processes
  - Academic and MPW service bureau access, with NDA
  - New, game changing access to SkyWater 130nm mixed signal fab

skywater

130nm Mixed Signal Process

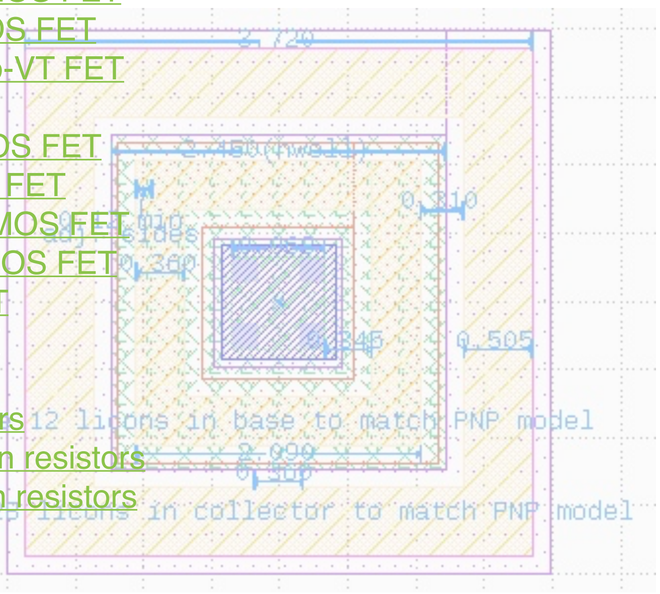
Not Just Low Tech



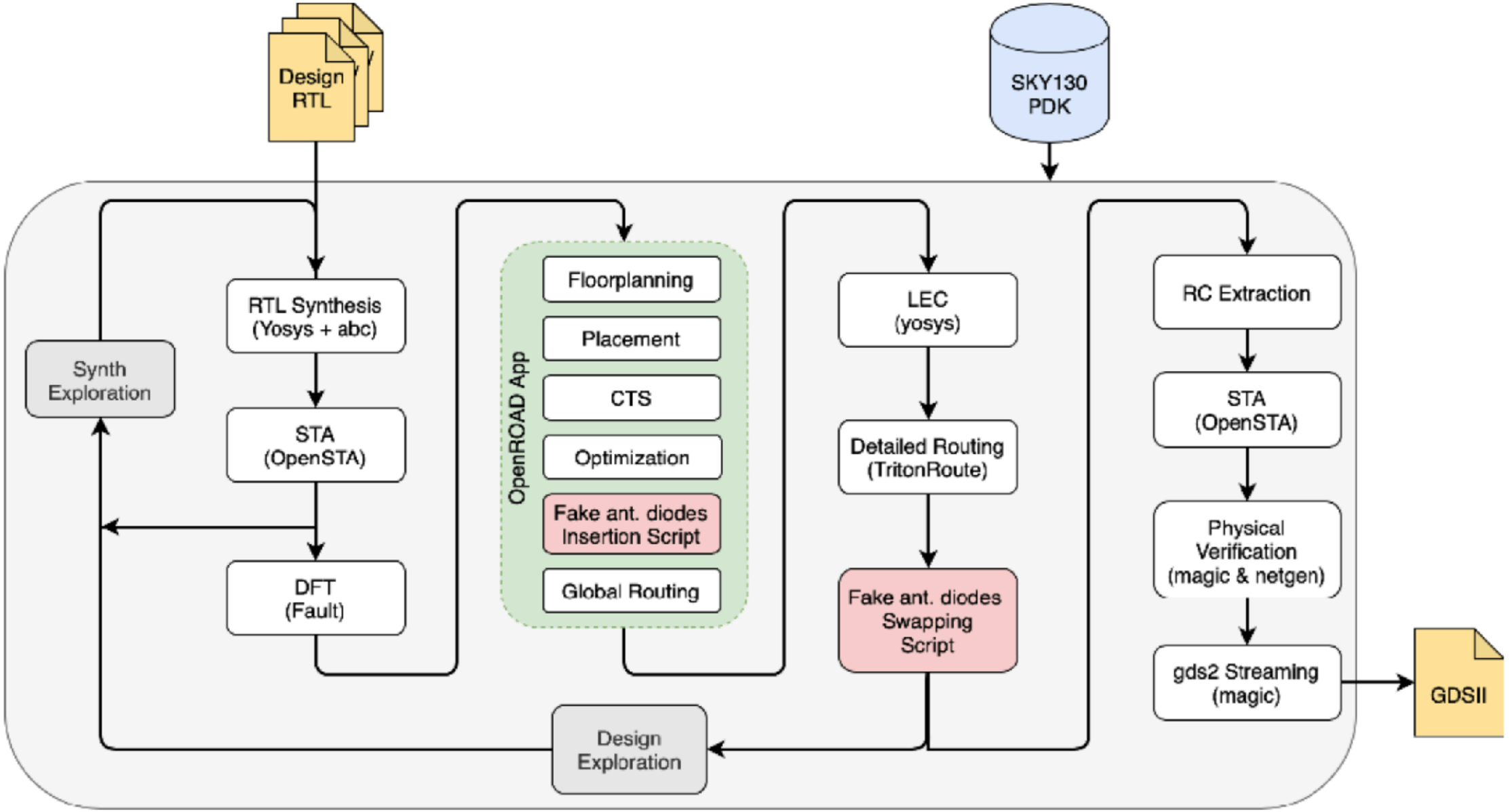


Device Details

- MiM Capacitor
- Varactors
- Vertical Parallel Plate (VPP) capacitors
- Diodes
- NMOS ESD FET
- 5.0V/10.5V NMOS FET
- 11V/16V NMOS FET
- 1.8V low-VT NMOS FET
- 1.8V NMOS FET
- 3.0V native NMOS FET
- 5.0V native NMOS FET
- 20V NMOS FET
- 20V isolated NMOS FET
- 20V native NMOS FET
- 20V NMOS zero-VT FET
- Bipolar (NPN)
- 5.0V/10.5V PMOS FET
- 10V/16V PMOS FET
- 1.8V high-VT PMOS FET
- 1.8V low-VT PMOS FET
- 1.8V PMOS FET
- 20V PMOS FET
- Bipolar (PNP)
- Generic Resistors
- P+ poly precision resistors
- P- poly precision resistors
- SONOS cells
- SRAM cells



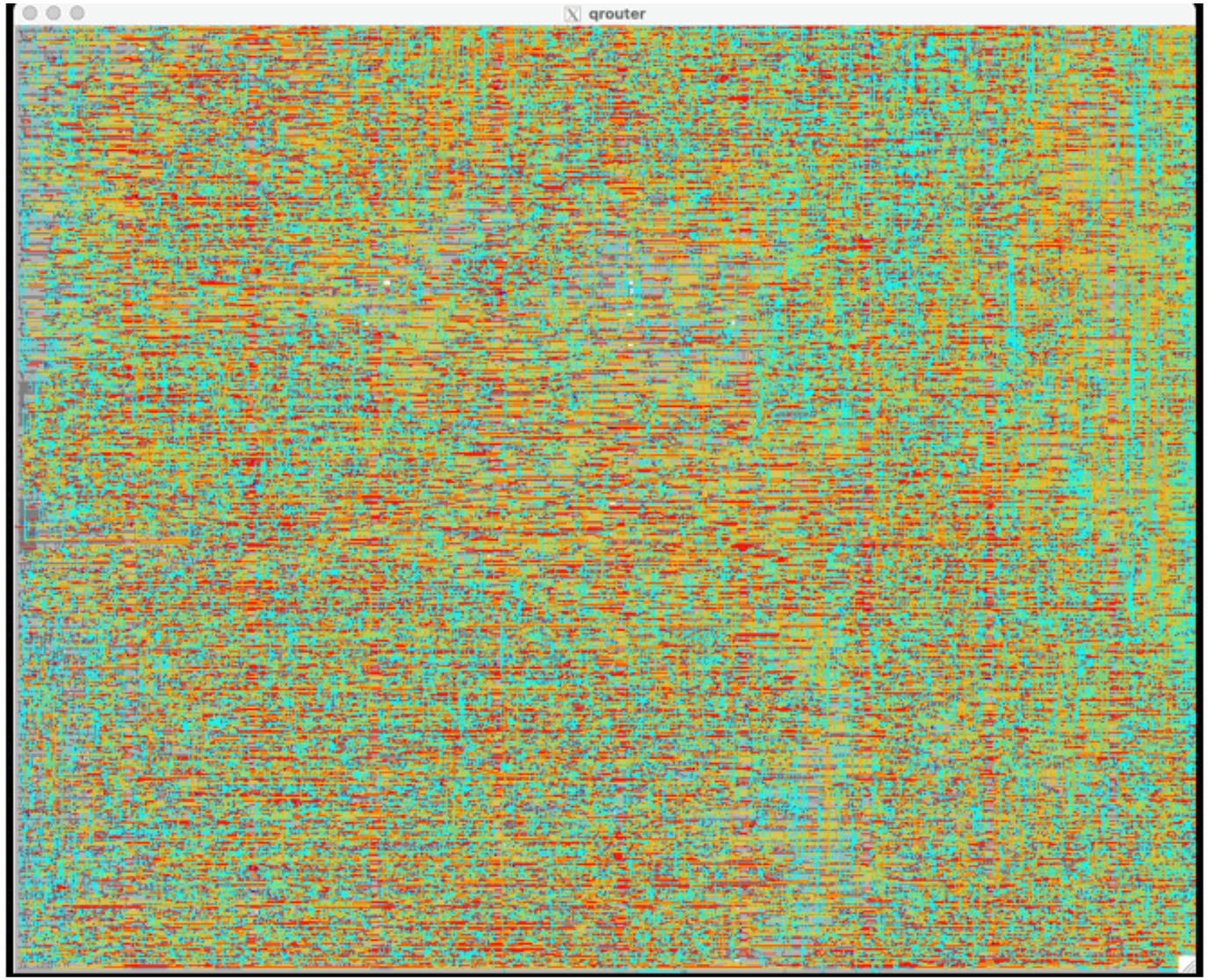
SkyWater130 OpenRoad Tools Flow



J1CPU 130nm QFlow Tools Place and Route

- Simple QFlow Tool Flow
- J1 CPU, first trial 492x461µm
  - 0.226mm^2
  - SkyWater 130nm hd cells

- Proof of Concept
  - Register file



4. Complete Design Process

1. Design Concept
2. Prototype using a standard or autogenerated SoC template
3. Simple interface hardware, and RTL for interface IP block
4. Develop and run software on the SoC in FPGA, with real interface HW
5. Synthesis, Place and Route with the same toolflow for standard cell ASIC
6. Verification and simulation
7. Tape out on a low cost process, hardware board design
8. Run proven software on new hardware with ASIC

Questions and Wrap Up

CORE SEMICONDUCTOR

Just the Start of the Story

Thanks

Please check out the J-Core GitHub
http://github.com/j-core

QFlow and OpenPDK
http://github.com/RTimothyEdwards

OpenRoad
http://github.com/The-OpenROAD-Project

J-core