ETH zürich

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Future
Computing
Laboratory

EFCL Winter School 2026, Track #1, Technical Talk #3:

Physical Design Is Not Push Button

Integrated Systems Laboratory (ETH Zürich)

Yichao Zhang yiczhang@iis.ee.ethz.ch

Diyou Shen dishen@iis.ee.ethz.ch

PULP Platform
Open Source Hardware, the way it should be!

@pulp_platform
pulp-platform.org
youtube.com/pulp_platform

Who we are?

Yichao Zhang

Ph.D. student in IC design
ETH Zurich, expect 2026.

Developer
Manycore Design, *PULP Platform.*

Lead Application Engineer,
Cadence, Singapore 2021.

Physical design engineer,
MediaTek Singapore, 2019.

Master's Degree in Electronics
NTU, Singapore, 2017.

Research Interests

Manycore Architecture

Scalar & Vector Parallel

Physically Feasible Design

B5G/6G PHY-layer Baseband

Ph.D. student in IC design
ETH Zurich, expect 2029.

Developer
Manycore Design, *PULP Platform.*

Master's Degree in ITET
ETH Zurich, 2021-2024.

Bachelor's Degree in EE
CU Boulder, 2017-2021.

Diyou Shen

Research Interests

Manycore Architecture

Scalar & Vector Parallel

Physically Feasible Design

B5G/6G RLC-Layer

Cache System

Fault-Tolerance

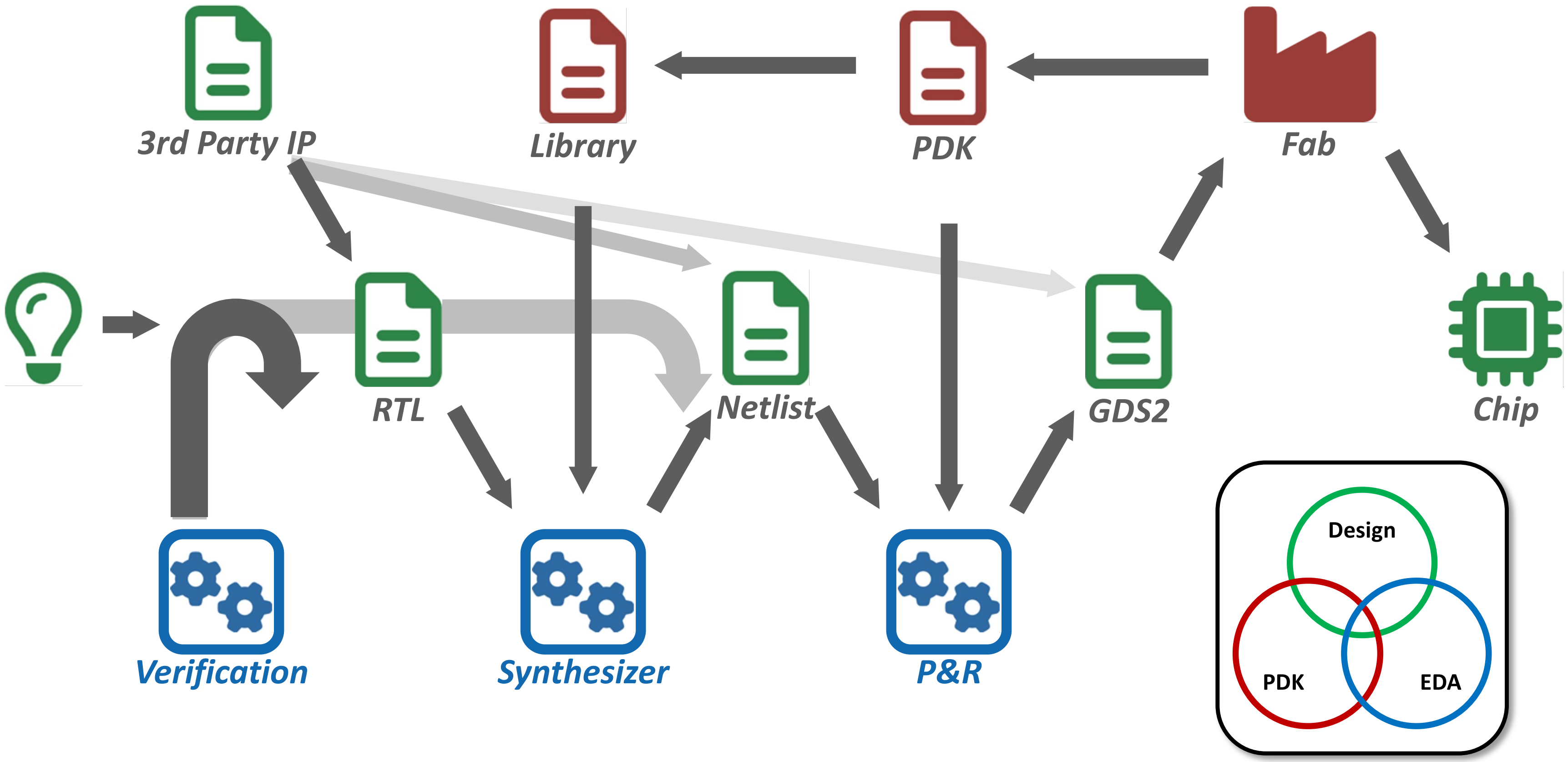
Today's main topic: **Physical Design**

Physical design is not an isolated stage

Physical design is not just playing EDA tools

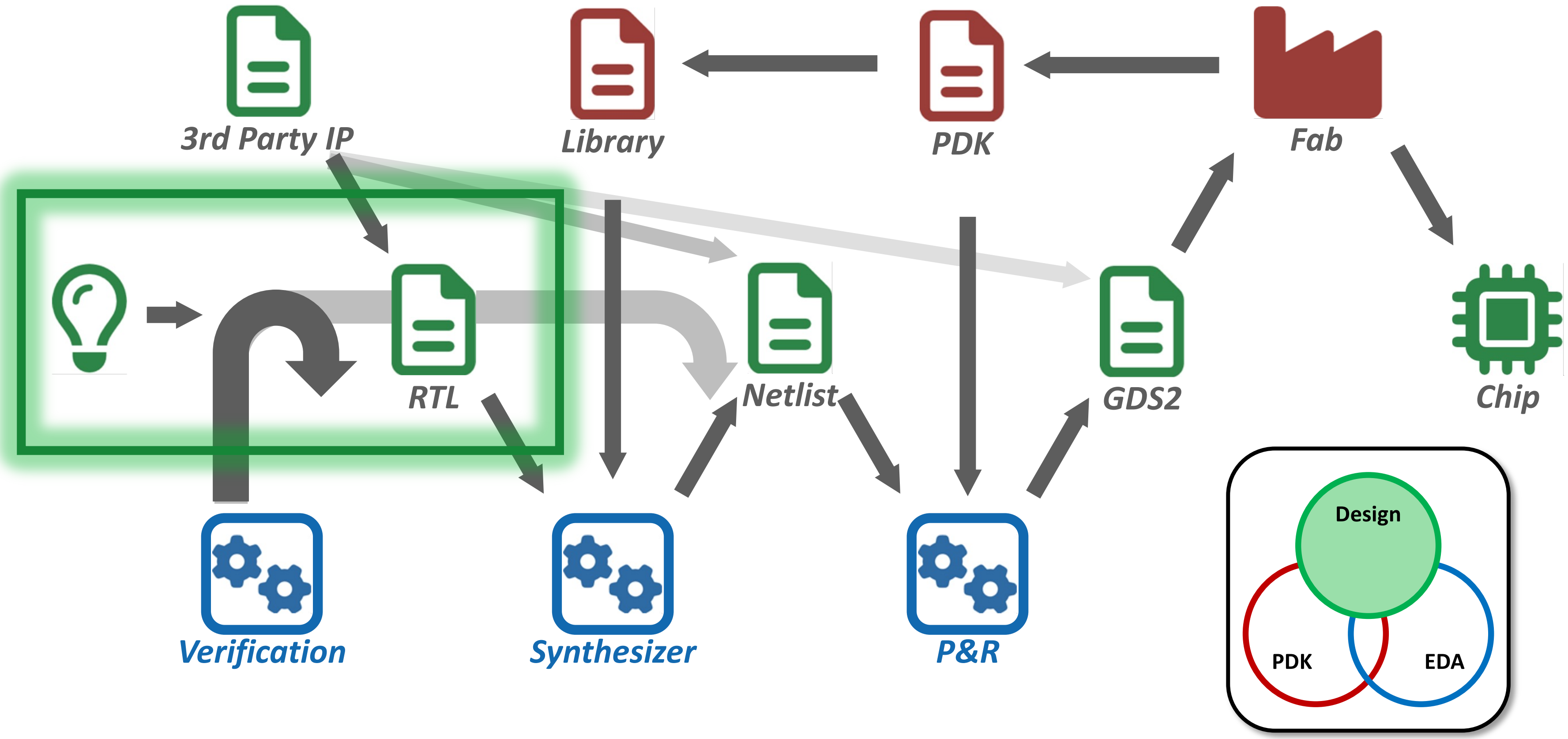
Physical design is not "push-button"
on an "I-have-no-idea" script

A simplified view of the IC design flow



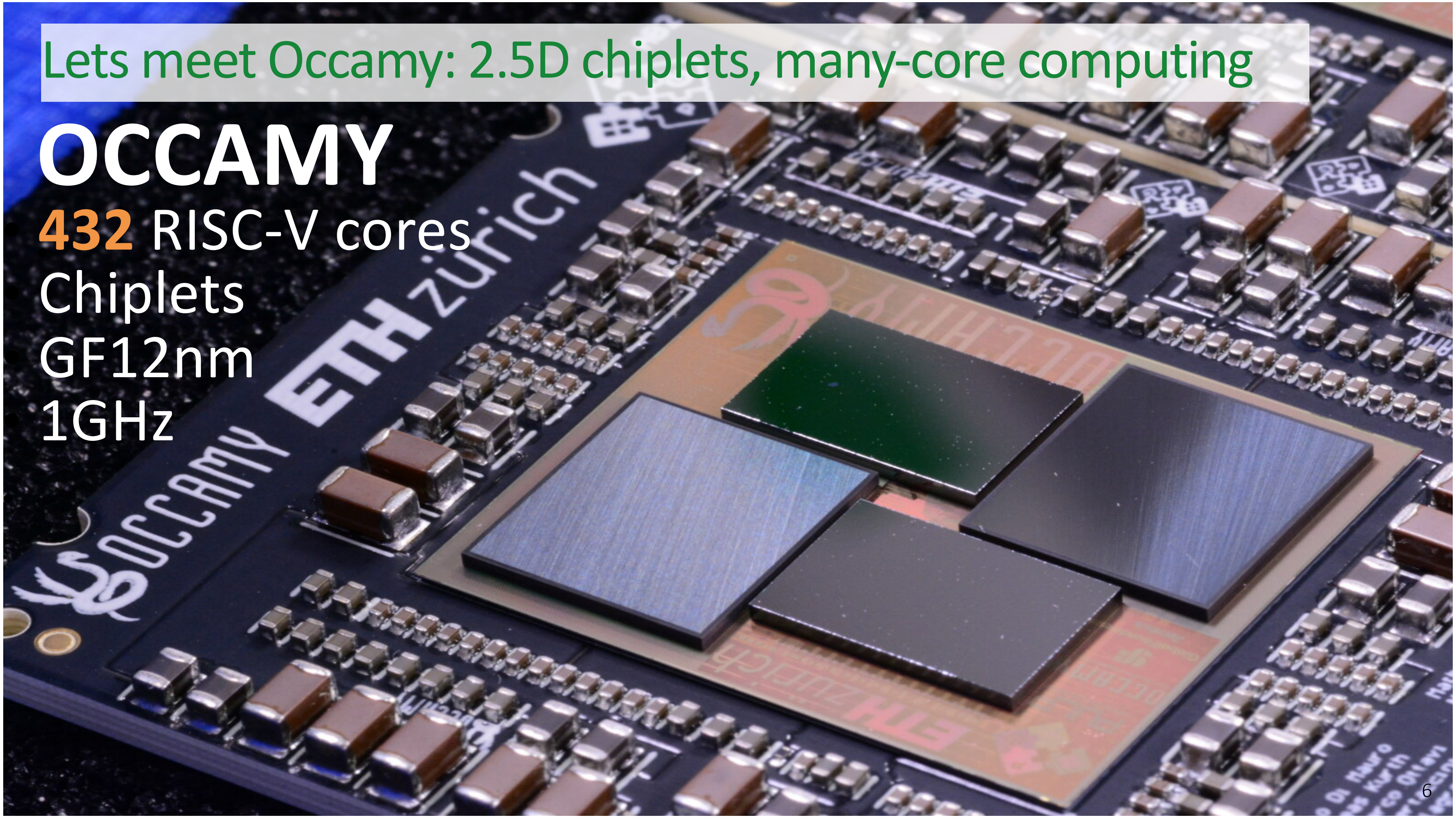
Icons taken from free icons from fontawesome.com

Before physical design: let’s see a ‘design.’



Icons taken from free icons from fontawesome.com

Lets meet Occamy: 2.5D chiplets, many-core computing



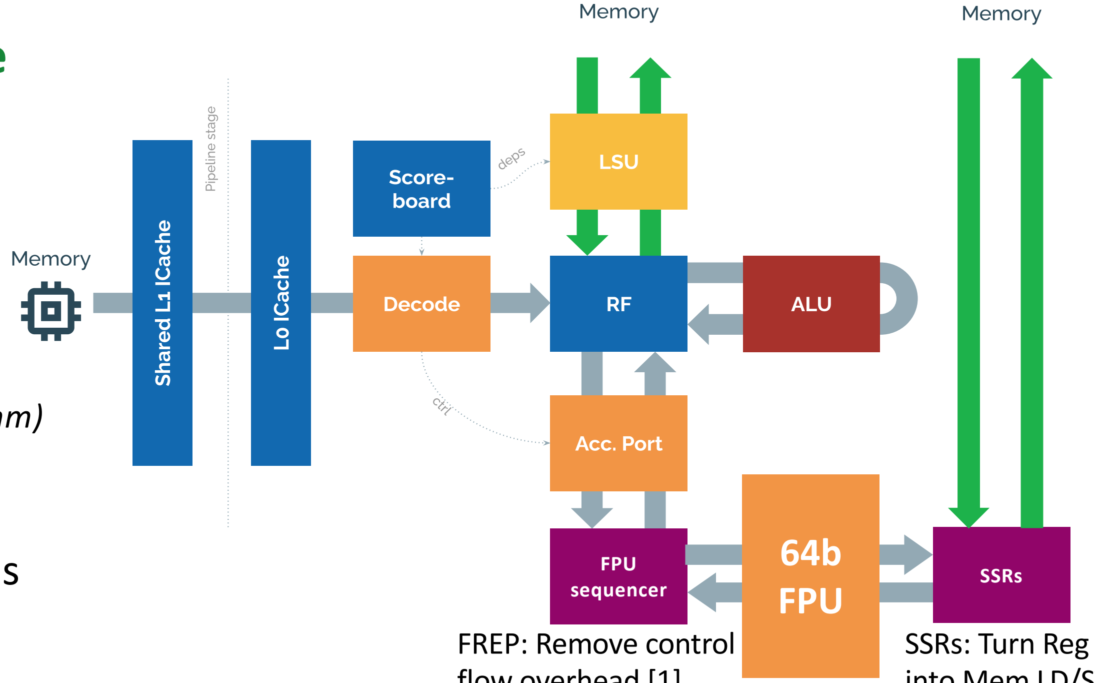
OCCAMY

- 432 RISC-V cores
- Chiplets
- GF12nm
- 1GHz

Level 0: we need a core

Snitch core: 32b, tiny, extendable

- **A simple RISC-V core:**
  - **Lightweight** microarchitecture
    - ~15-25 kGE
  - **Latency tolerant**
  - Competitive **frequency** *(>1GHz, GF12nm)*
- **Extensibility:**
  - Performance through ISA extensions
  - Capable 64b FPU with many extensions



FREP: Remove control flow overhead [1]

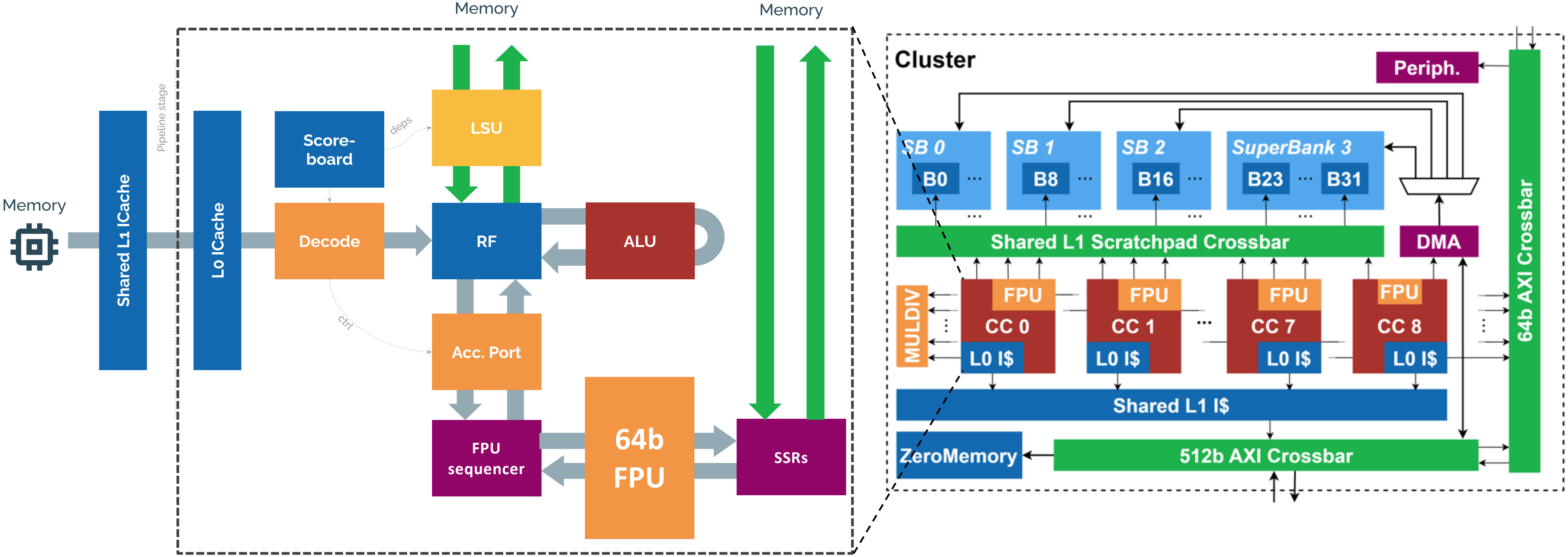
SSRs: Turn Reg R/W into Mem LD/ST [2]

[1] F. Zaruba et al., "Snitch: A Tiny Pseudo Dual-Issue Processor for Area and Energy Efficient Execution of Floating-Point Intensive Workloads," in IEEE Transactions on Computers, vol. 70, no. 11, pp. 1845-1860, 1 Nov. 2021, doi: 10.1109/TC.2020.3027900.

[2] F. Schuiki et al., "Stream Semantic Registers: A Lightweight RISC-V ISA Extension Achieving Full Compute Utilization in Single-Issue Cores," in *IEEE Transactions on Computers*, vol. 70, no. 2, pp. 212-227, 1 Feb. 2021, doi: 10.1109/TC.2020.2987314.

Level 1: scale-up to multi-core cluster

Cluster = cores + interconnect + shared-L1-memory



Occamy cluster – 5 MGE

[3] P. Scheffler et al., "Indirection Stream Semantic Register Architecture for Efficient Sparse-Dense Linear Algebra," 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2021, pp. 1787-1792, doi: 10.23919/DATE51398.2021.9474230.

[4] L. Bertaccini et al., "MiniFloat-NN and ExSdotp: An ISA Extension and a Modular Open Hardware Unit for Low-Precision Training on RISC-V Cores," *2022 IEEE 29th Symposium on Computer Arithmetic (ARITH)*, Lyon, France, 2022, pp. 1-8, doi: 10.1109/ARITH54963.2022.00010.

8 Snitch compute cores

- Single-stage, small Integer control core

9th Core: DMA

- 512 bit data interface
- HW support to autonomously copy 2D shapes
- Higher-dimensionality can be handled by SW

128 kB TCDM

- Scratchpad for predictable memory accesses
- 32 Banks

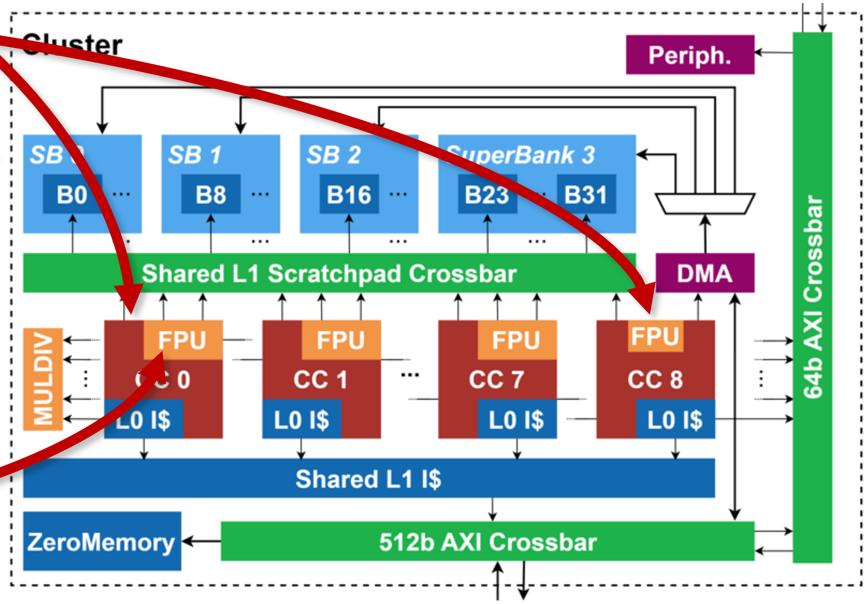
Custom ISA extensions

- Xfrep, Xssr
- New: Xissr sparsity support [3]

1 FPU per Snitch core

- Decoupled and heavily pipelined
- Multi-format FPU (+SIMD)
- New: Minifloat support + SDOTP [4]

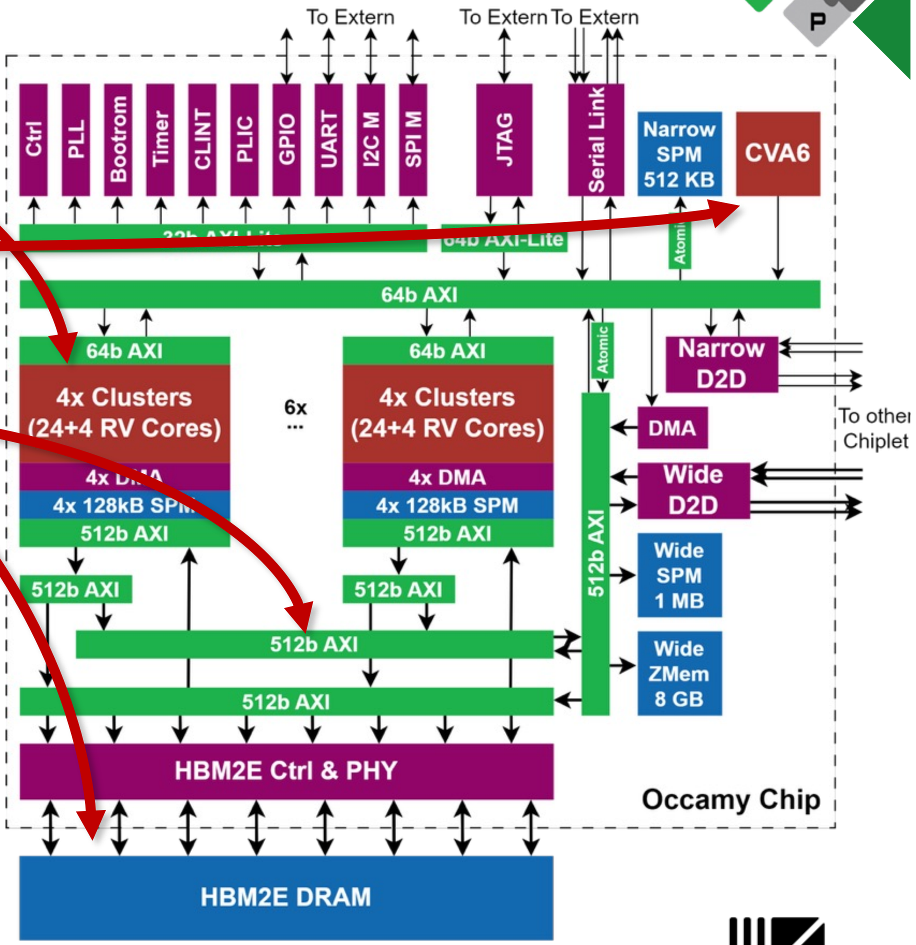
8GB/s in each direction



64GB/s in each direction

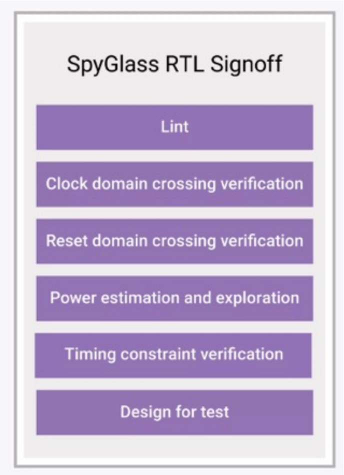
Level 2 (and higher): scale-out to multi-cluster

- **4 clusters per "quadrant"**
  - (8+1) cores per cluster
- **Totally 6 quadrants + 1 host core per die**
- **8-channel HBM2e (16GB)**
- **Multi-level AXI crossbar interconnect**
  - Cluster to cluster
  - Main memory to cluster (DMA, HBM)
  - Peripherals
  - Die to die interface (serial link)



RTL done. Is the time moving to physical design?

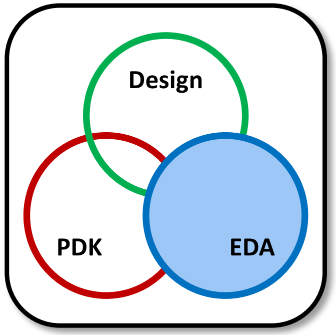
- **NO! RTL signoff is super important!**
- **Don’t trust your “experiences”. EDA tool can be your “spyglass”:**
  - Syntax checks
  - Semantic checks
  - Synthesizability checks
  - … …



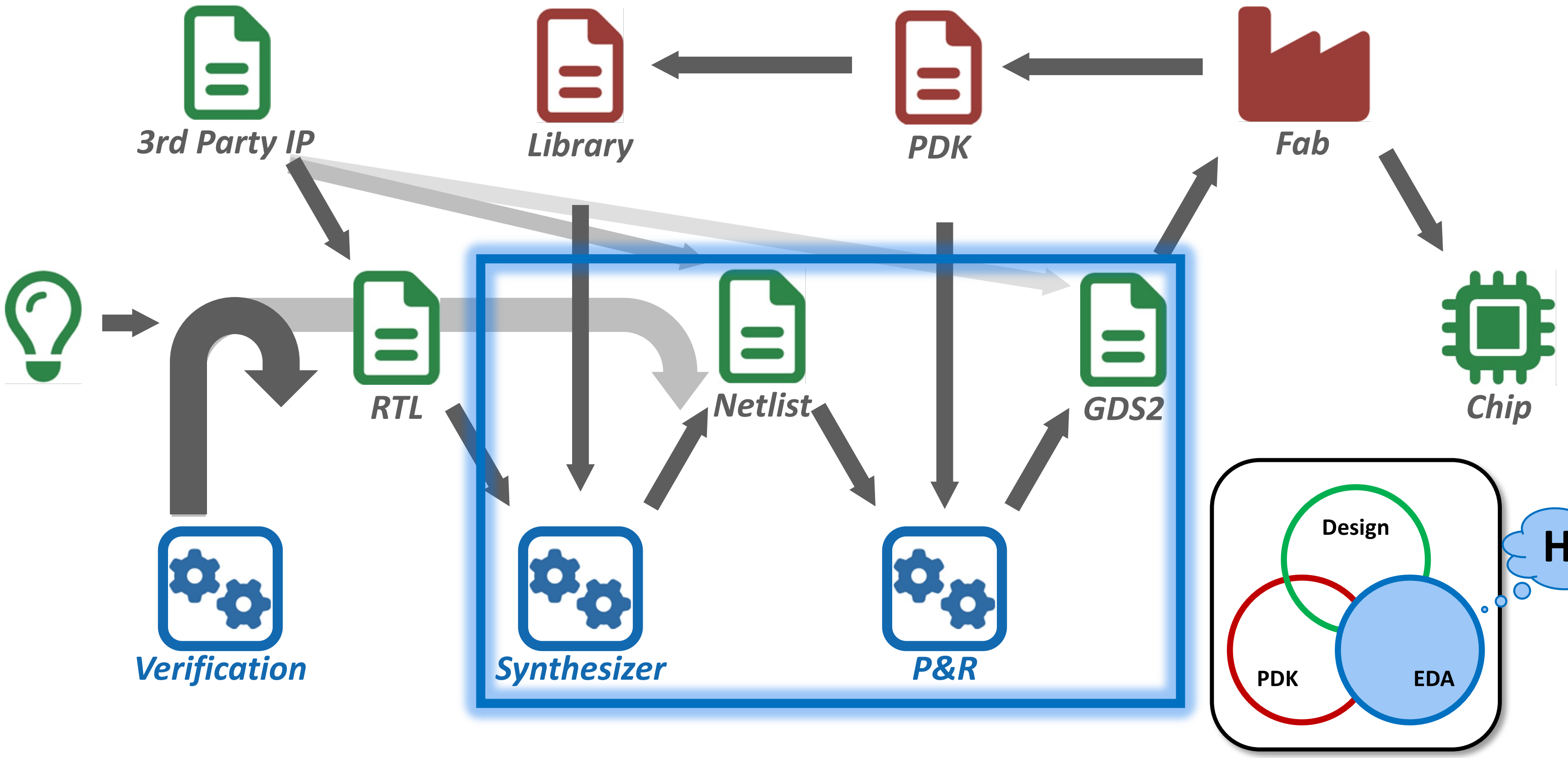
SpyGlass RTL Signoff

- Lint
- Clock domain crossing verification
- Reset domain crossing verification
- Power estimation and exploration
- Timing constraint verification
- Design for test

- **We faced many times that the PnR tool stuck in/infinite loop/crashed. It’s not the PnR tool’s fault**
  - Combinational loop
  - Non-resettable register
  - … …

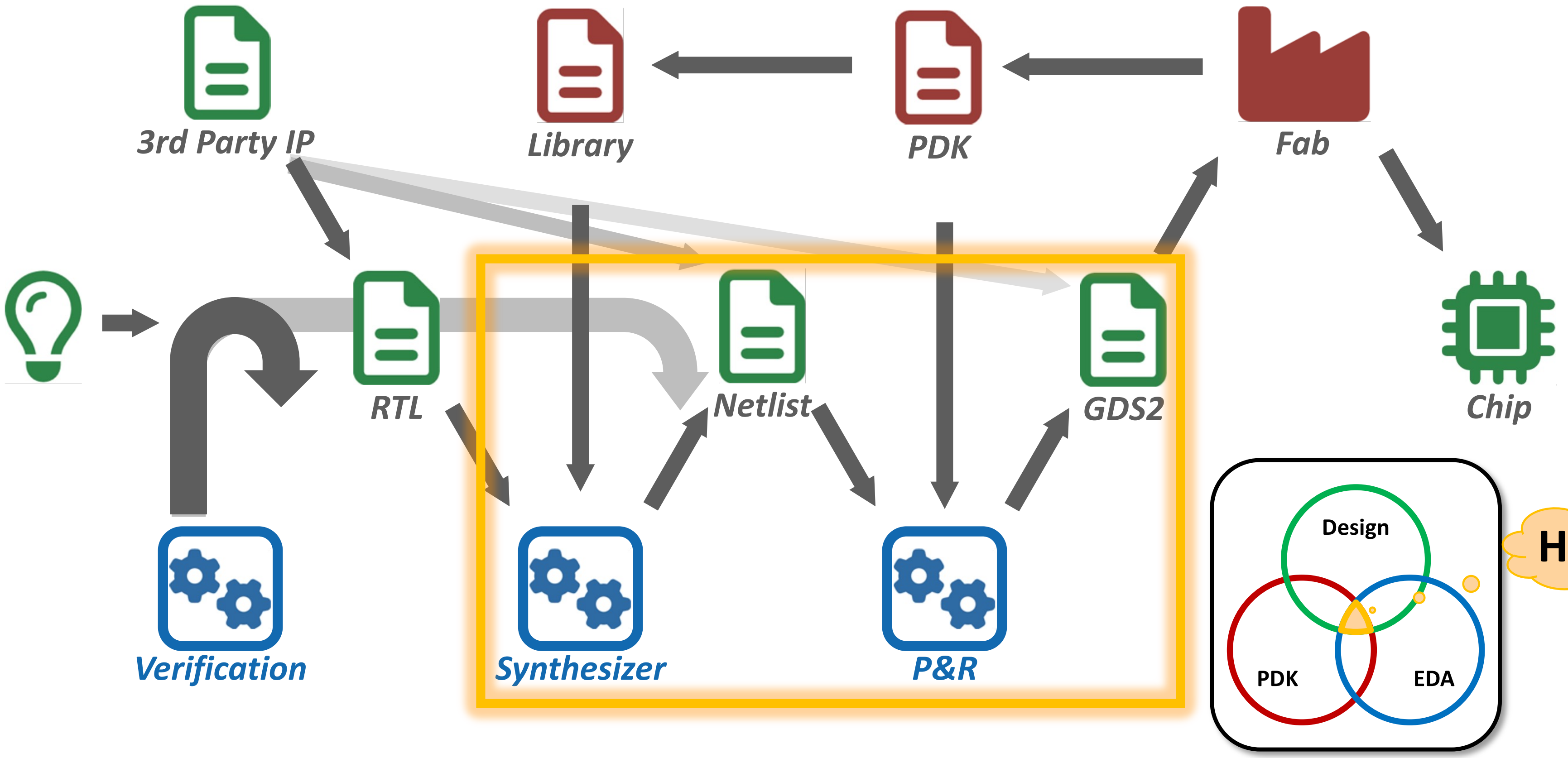


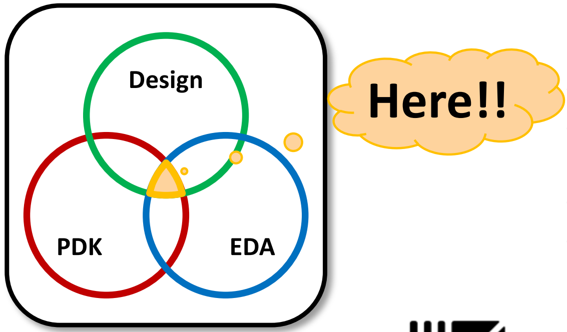
Now let’s move to the physical design



Icons taken from free icons from fontawesome.com

Now let's move to the physical design

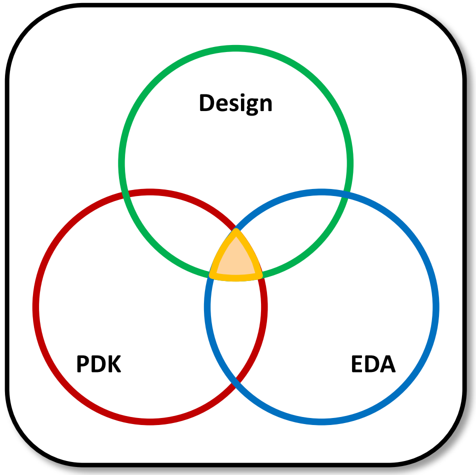




Icons taken from free icons from fontawesome.com

Physical design is not an isolated stage

- **You need to be an "expert" for:**
  - Your design (e.g. interco. wiring complexity)
  - PnR tool (user guide is your bible)
  - PDK (how much physical resource you have)
- **Example: we faced two key challenges in Occamy**
  - Is your computing cluster design a good PPA?
  - Is your interconnect physically feasible? *("pen+paper" is your good friend)*

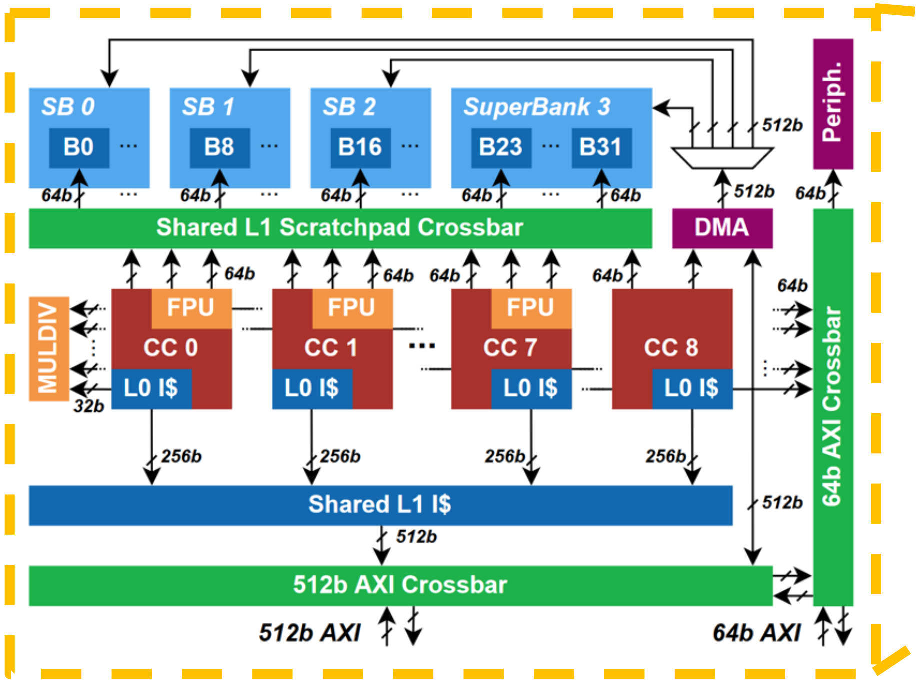


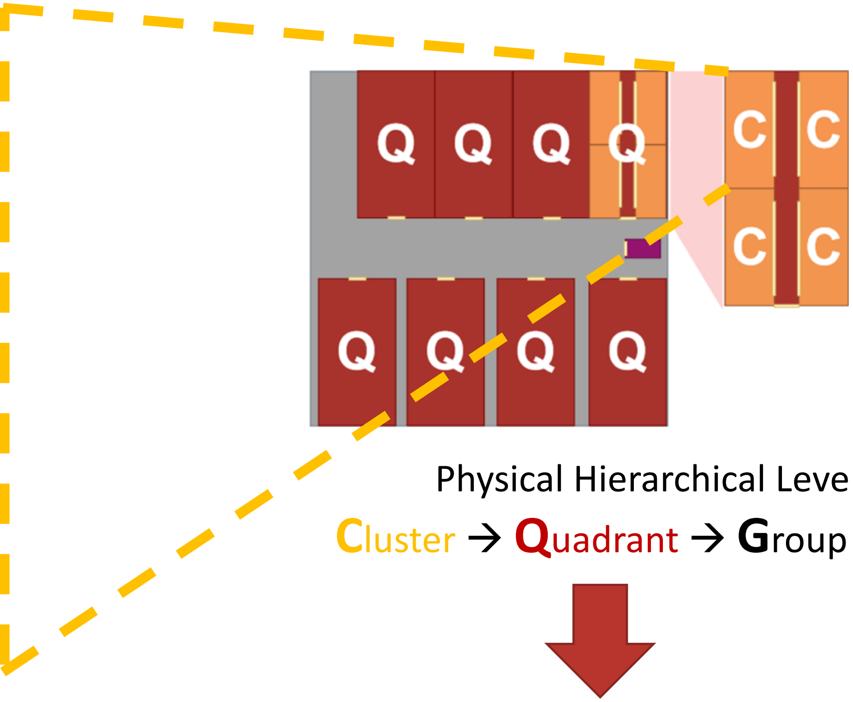
Next, we will see a cluster **floorplan study**[5]

(Floorplan is not a video game)

[5] G. Paulin, M. Cavalcante, P. Scheffler, L. Bertaccini, Y. Zhang, F. Gürkaynak, and L. Benini, "Soft Tiles: Capturing Physical Implementation Flexibility for Tightly-Coupled Parallel Processing Clusters", in Computer Society Annual Symposium on VLSI (ISVLSI), IEEE, Nicosia, Cyprus, Jul. 2022, pp. 44–49.

Cluster scaling out is also a physical matter



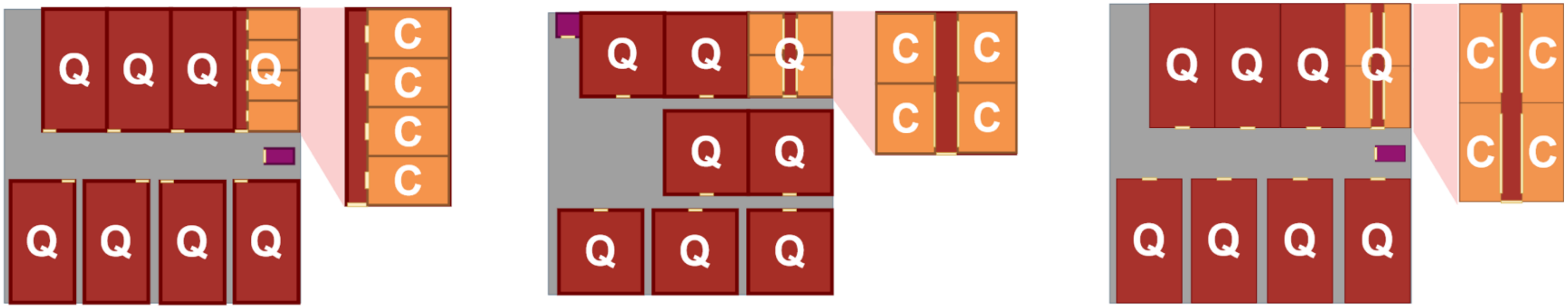


Physical Hierarchical Level:

Cluster → Quadrant → Group → … …

Cluster tile can be **replicated** to build a **scaled-out high-performance acceleration system**

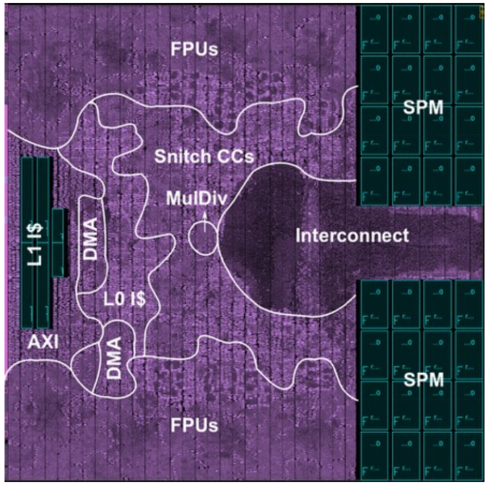
Floorplan layout and aspect ratio



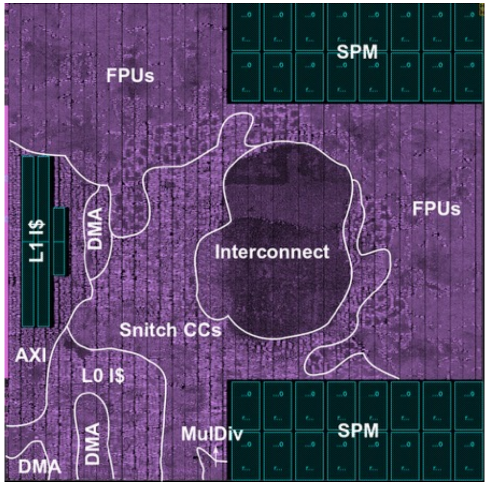
- **Die size and shape set your first floorplan constraints**
- **Which layout and aspect ratio to choose?**
  - Hard to know before implementing the design
  - Many constraints on the top-level (routing, position of hard IPs, system area...)
- **Cluster is your computing block; its PPA should be placed as the first priority**

Lets talk about the macro placement

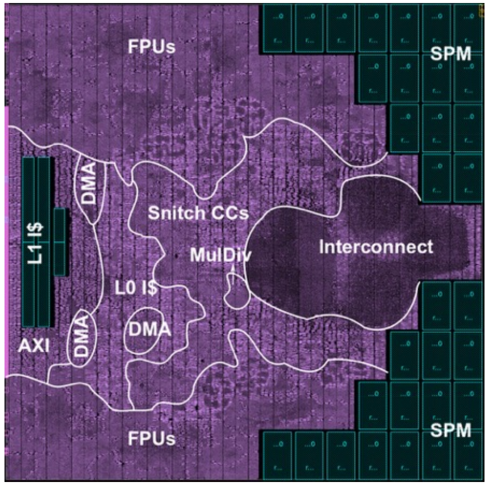
- First, let’s fix the aspect ratio (square-shaped, 1:1)
  - 9 cores tightly coupled to 32 SPM banks → **floorplan is not evident**
- We came up with three generic "floorplan styles" to implement the cluster



One-sided



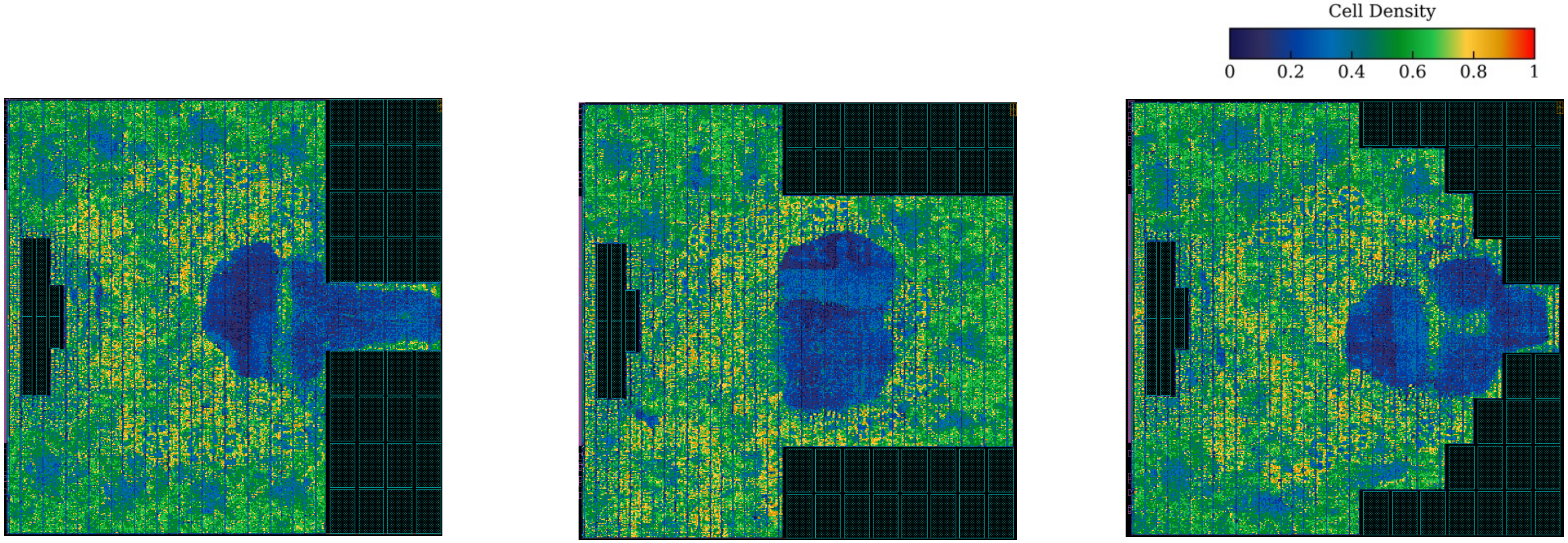
Two-sided



U-Shaped

Macro placement → module placement

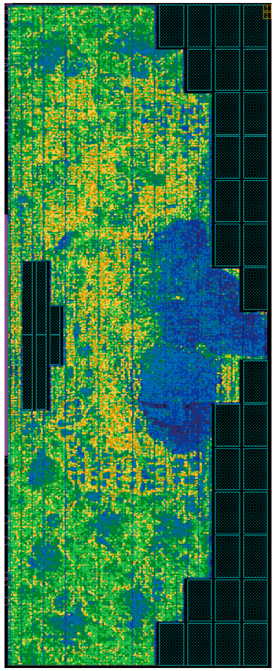
- Despite an overall cell density of 60%, the large interconnect means that the cell density peaks at 80% in some areas
- Interconnect placement is key!



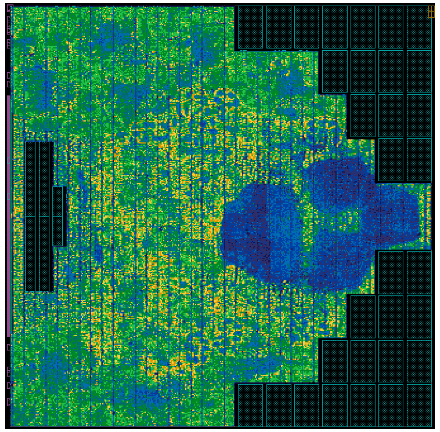
One-sided **Two-sided** **U-Shaped**

Floorplan shape: aspect ratio

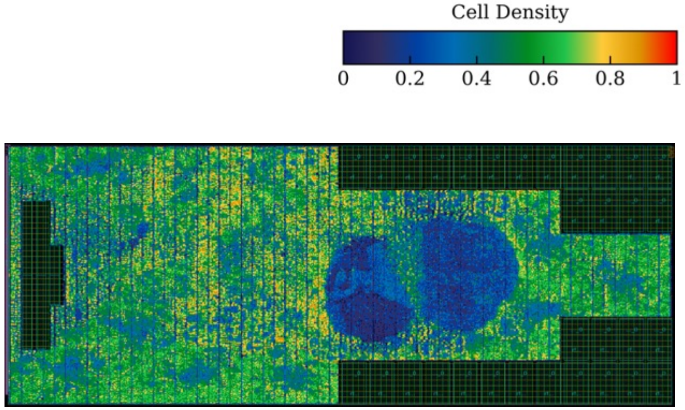
- An example with the U-shaped macro placement:
  - Changes in the **aspect ratio**, keeping the same area



Tall (1:2.5)

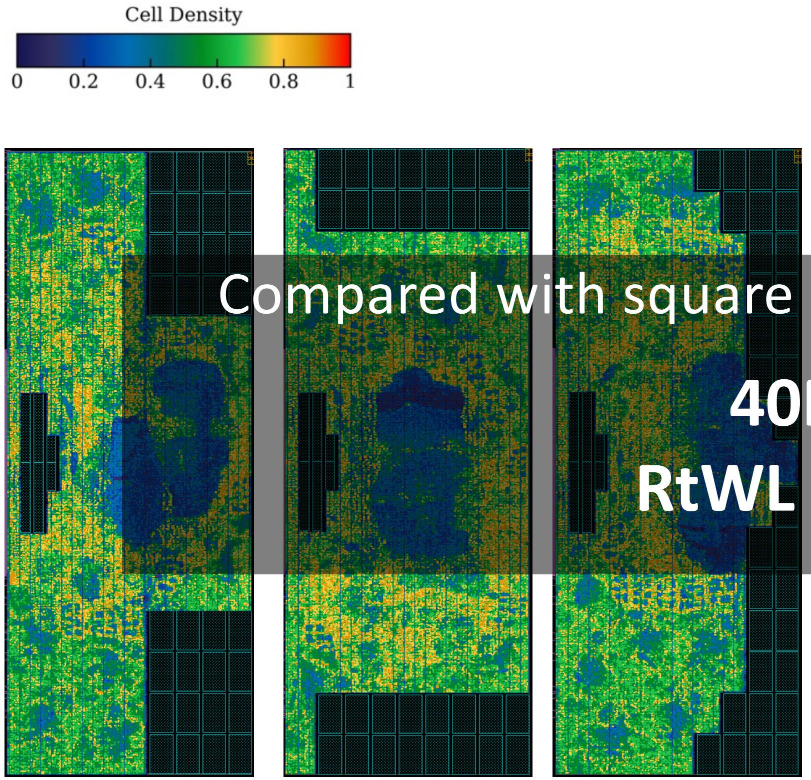


Square (1:1)

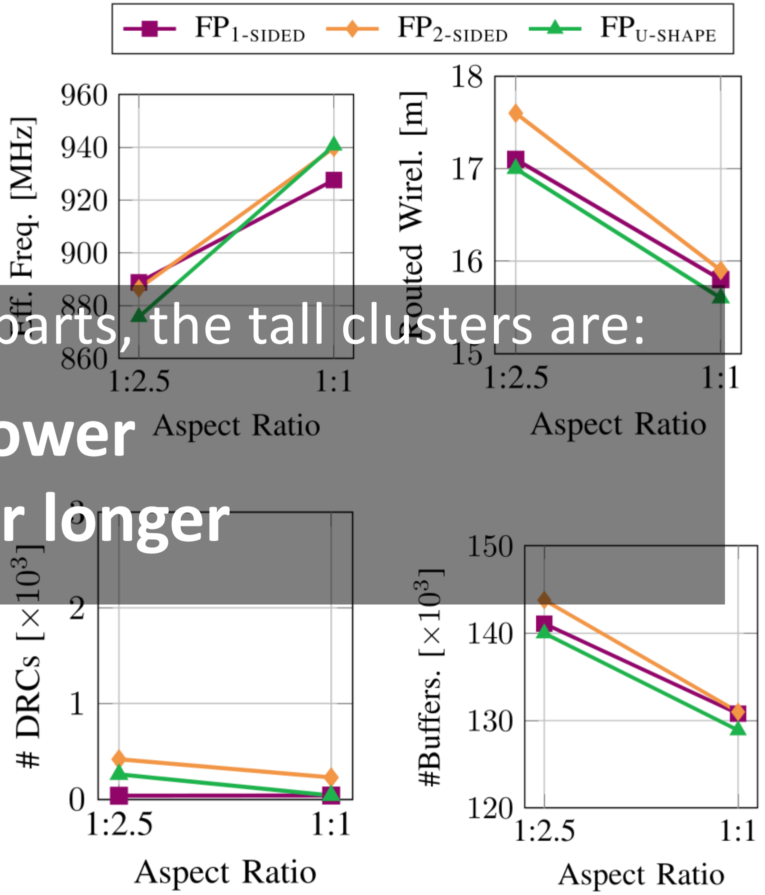


Wide (2.5:1)

What about the “Tall” shape?



One-sided **Two-sided** **U-Shaped**

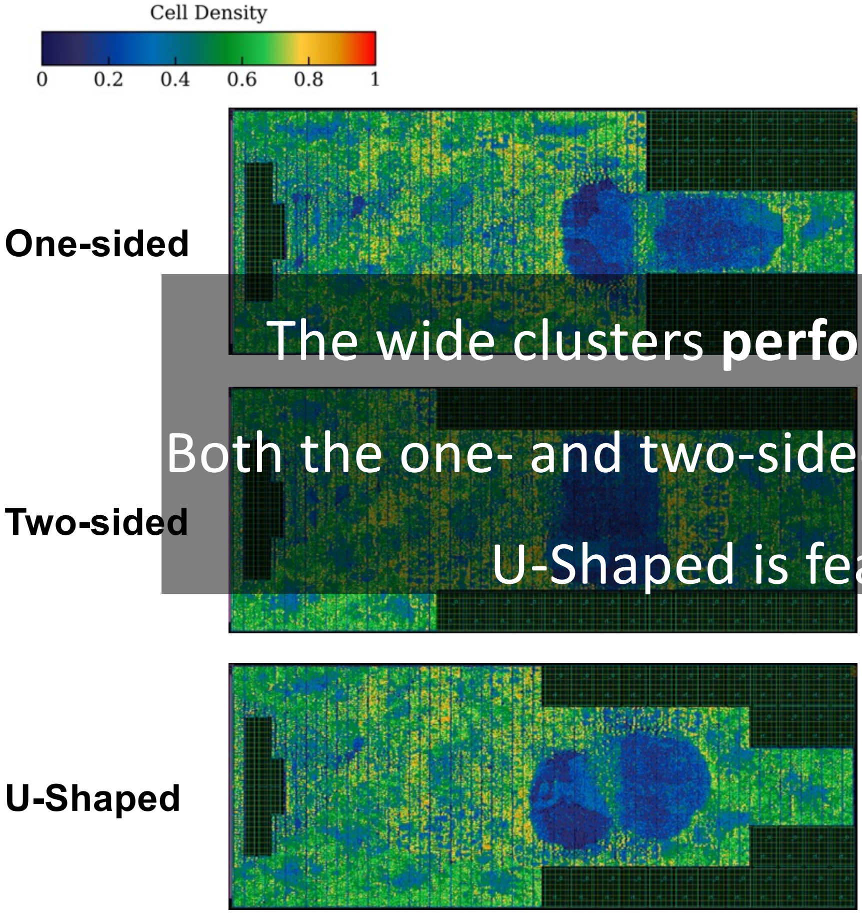


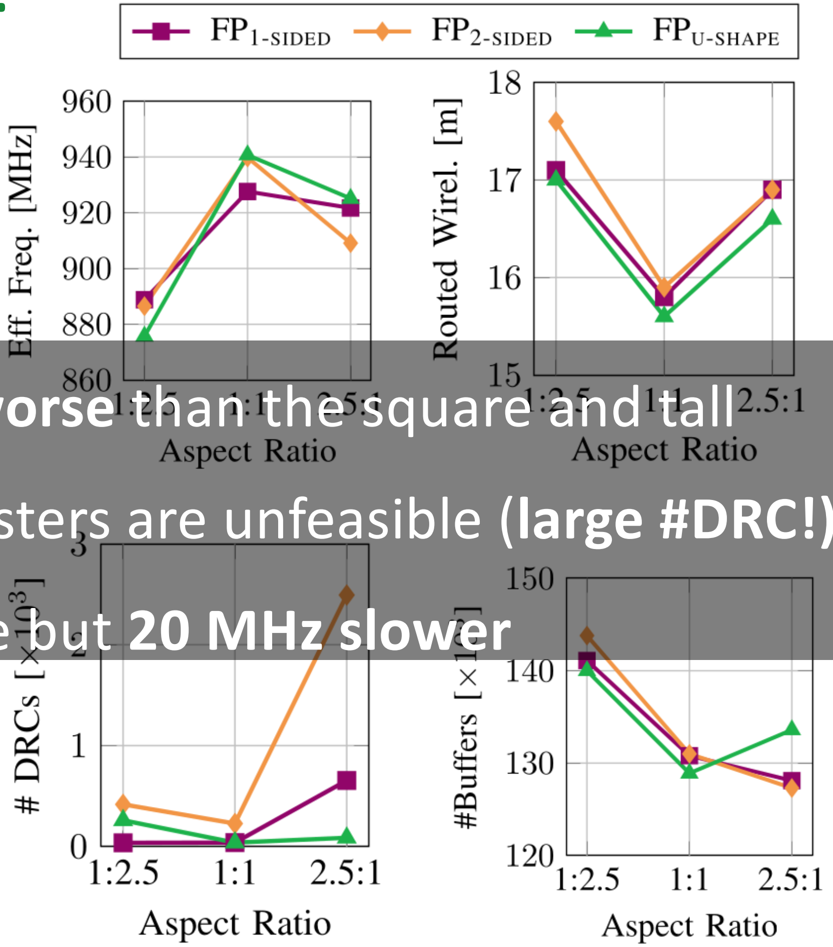
Compared with square counterparts, the tall clusters are:

40MHz slower

RtWL 1 meter longer

What about the “Wide” shape?





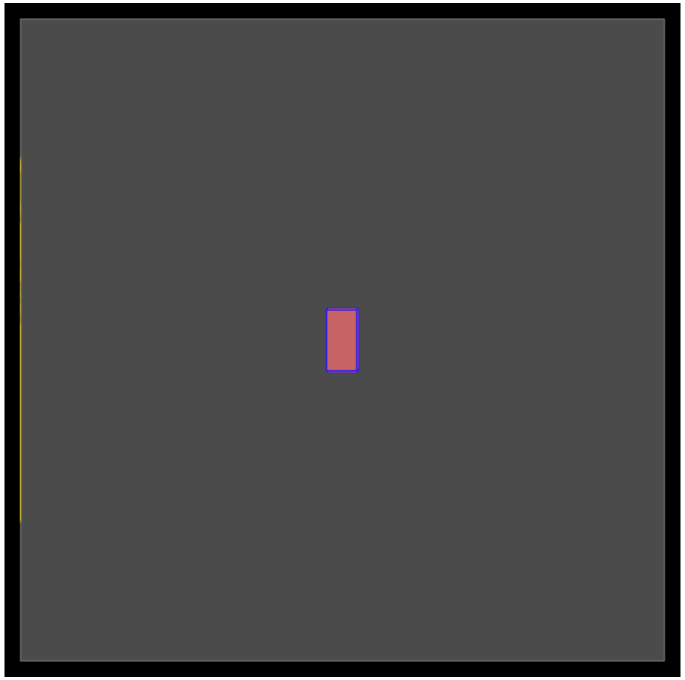
The wide clusters **perform worse** than the square and tall

Both the one- and two-sided clusters are unfeasible (**large #DRC!**)

U-Shaped is feasible but **20 MHz slower**

Future, could AI help?

- **In our design, U-shape + Square wins**
  - Similar floorplan, but results change A LOT
- **Design will continue to go UP, an example:**
  - MemPool/TeraPool in Pulp, 256/1024 cores shared-L1 single cluster.
  - Base-implement block = 64 cores + 256 SPM banks
- **AI-enhanced physical design?**
  - MemPool by AutoDMP with AI and Nvidia GPUs
  - Still a lot of open questions; we have on-going projects working on it; keep your eyes on us

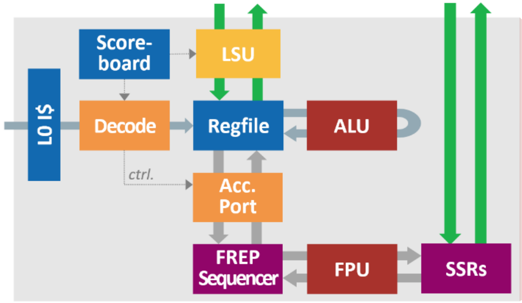


Reference: https://developer.nvidia.com/blog/autodmp-optimizes-macro-placement-for-chip-design-with-ai-and-gpus/

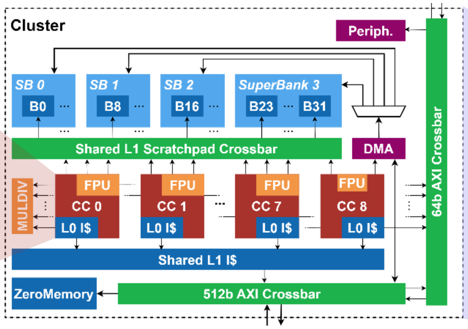
Achieving Scale through Hierarchical Design

Tall clusters with
U-Shaped floorplan

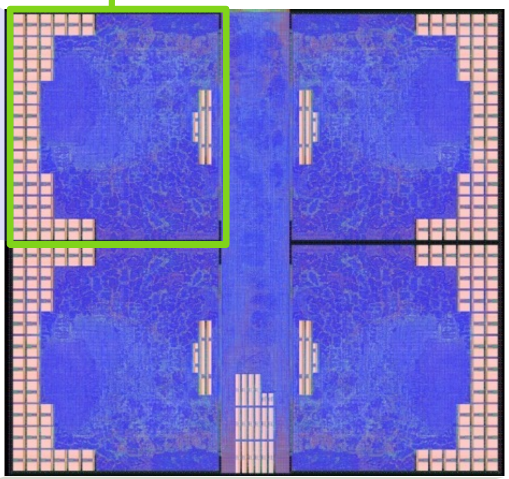
Snitch Core



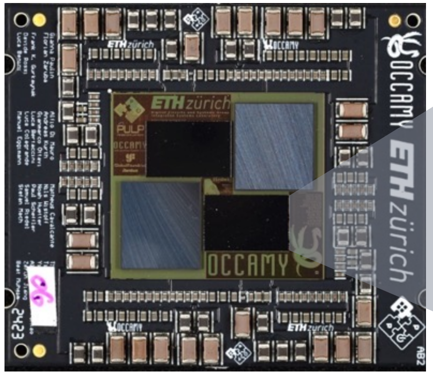
Snitch Cluster



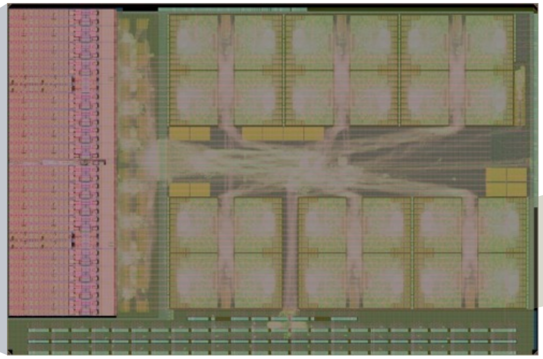
Occamy Quadrant



Occamy System



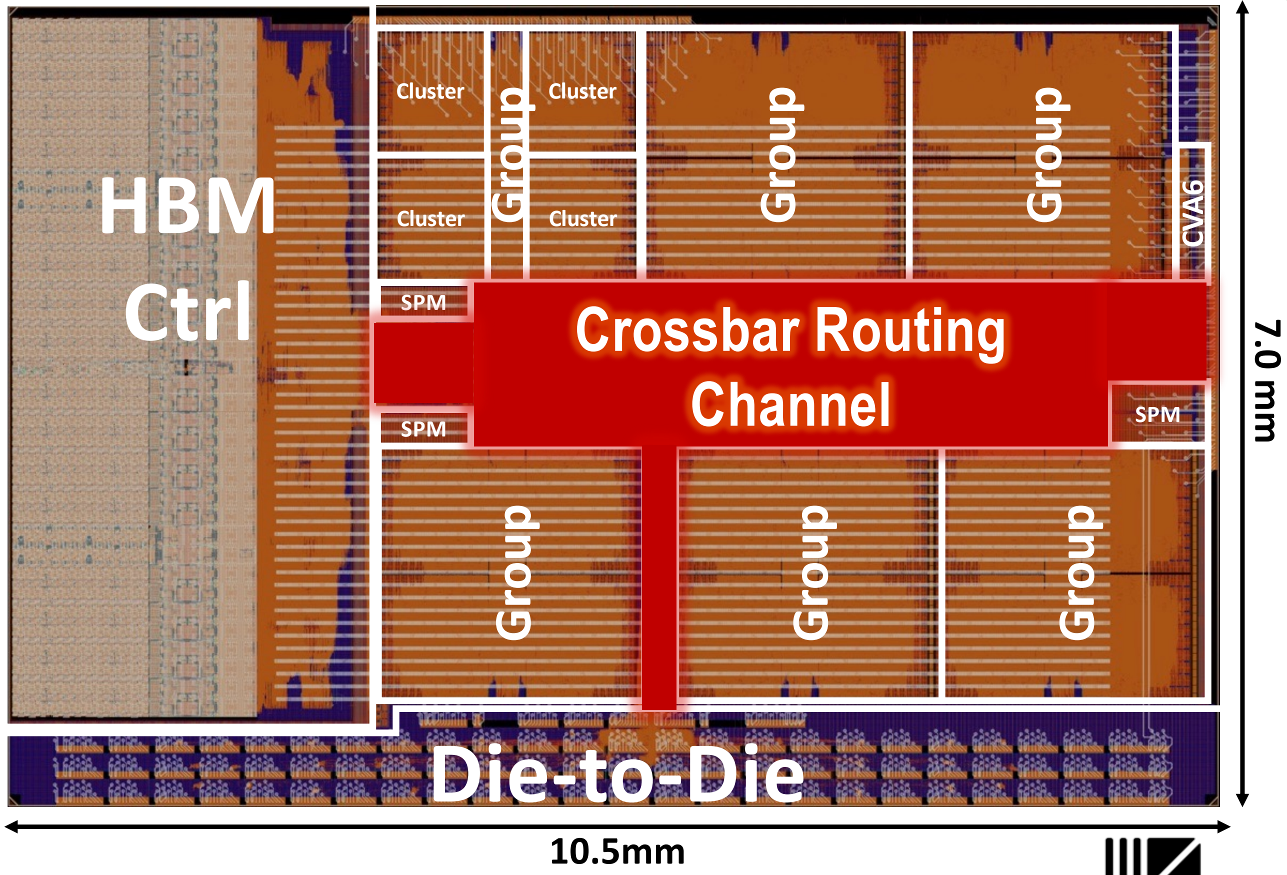
Occamy Chiplet



Soft-tiles gave us a headstart on what we could do and which floorplans to use to make Occamy's top-level feasible

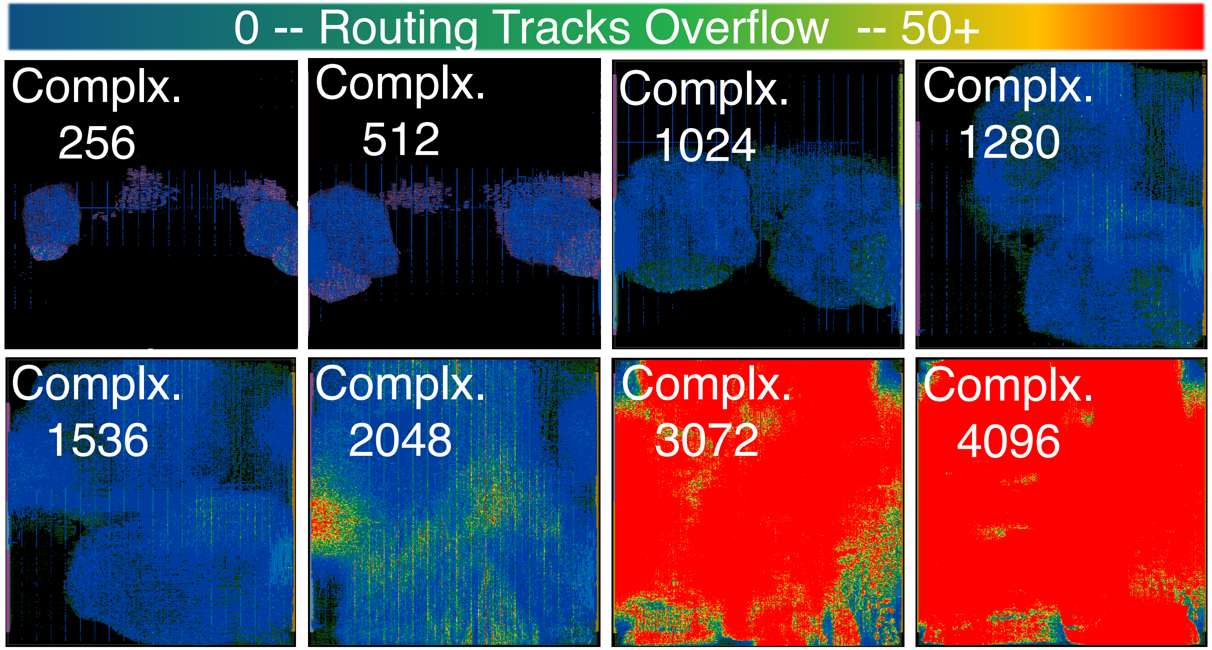
Is occamy perfect enough?

- **Crossbar Interconnect:**
  - Low-latency, high BW, but…
  - Quadratically increased wiring complexity (MxN)
  - Congestion-driven FP design: large routing channel (die area)



Is occamy perfect enough?

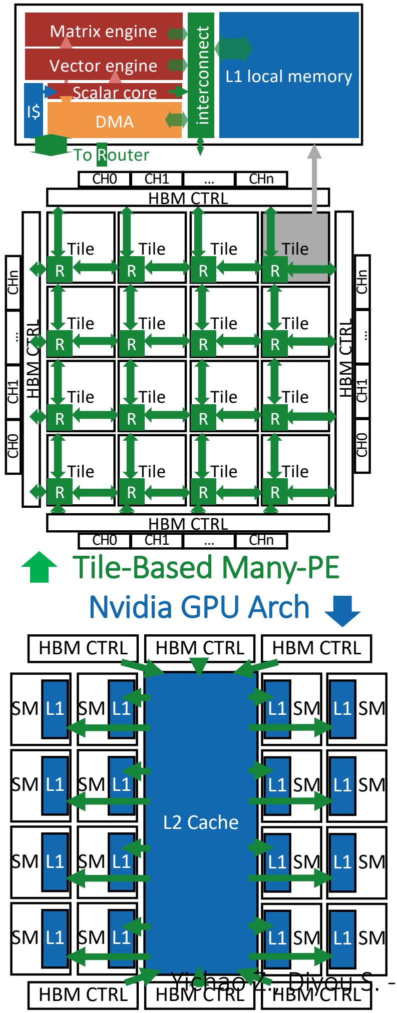
- **Crossbar Interconnect:**
  - Low-latency, high BW, but…
  - Quadratically increased wiring complexity (MxN)
  - Congestion-driven FP design: large routing channel (die area)
- **You need to think before PnR:**
  - Available routing resource
  - Max. wiring complex
- **Hierarchically splitted Xbars**
  - Trade-offs bw BW - Routing

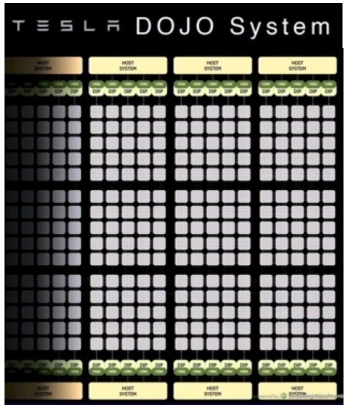


Routing Congestion of Logarithmic-Crossbar-Based Interconnect at Different Complexities (GF12nm, 13M).

Cannot do better? Go back to think the design

- **Tile-based many-PE accelerators**
  - Structured as meshes of compute tiles
    - Matrix, vector and scalar engines
    - Local explicitly managed memories
  - **Interconnect with scalable NoC topology**
  - Main memory positioned at die boundaries
- Favors **silicon efficiency** and **scalability**
- Examples
  - Tesla's Dojo, Tenstorrent's Blackhole



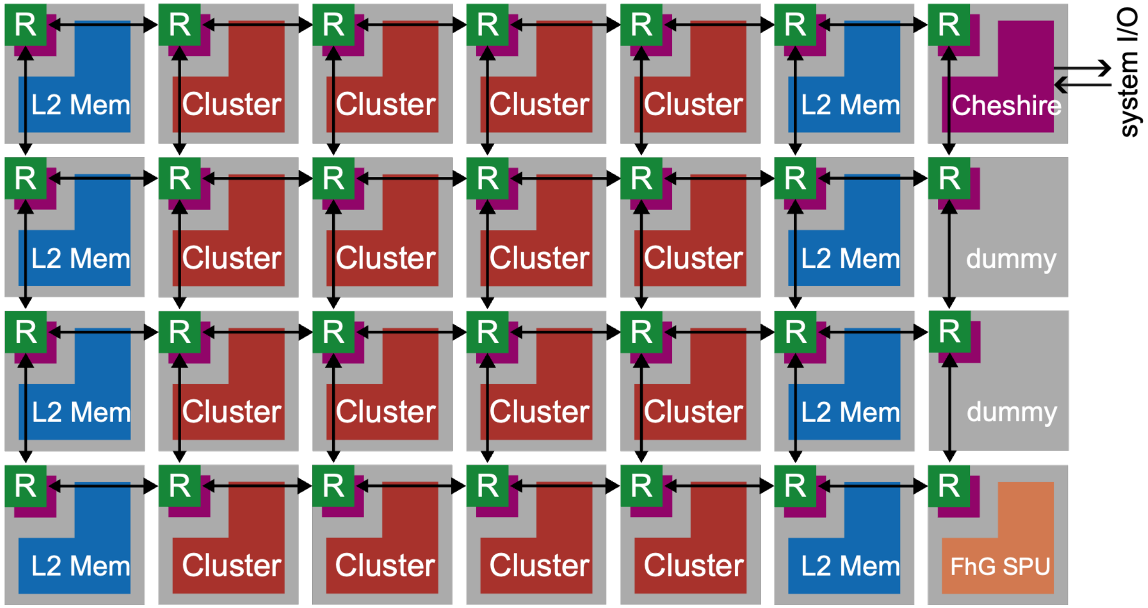


tenstorrent

From Occamy to Picobello: further scale in 7nm TSMC

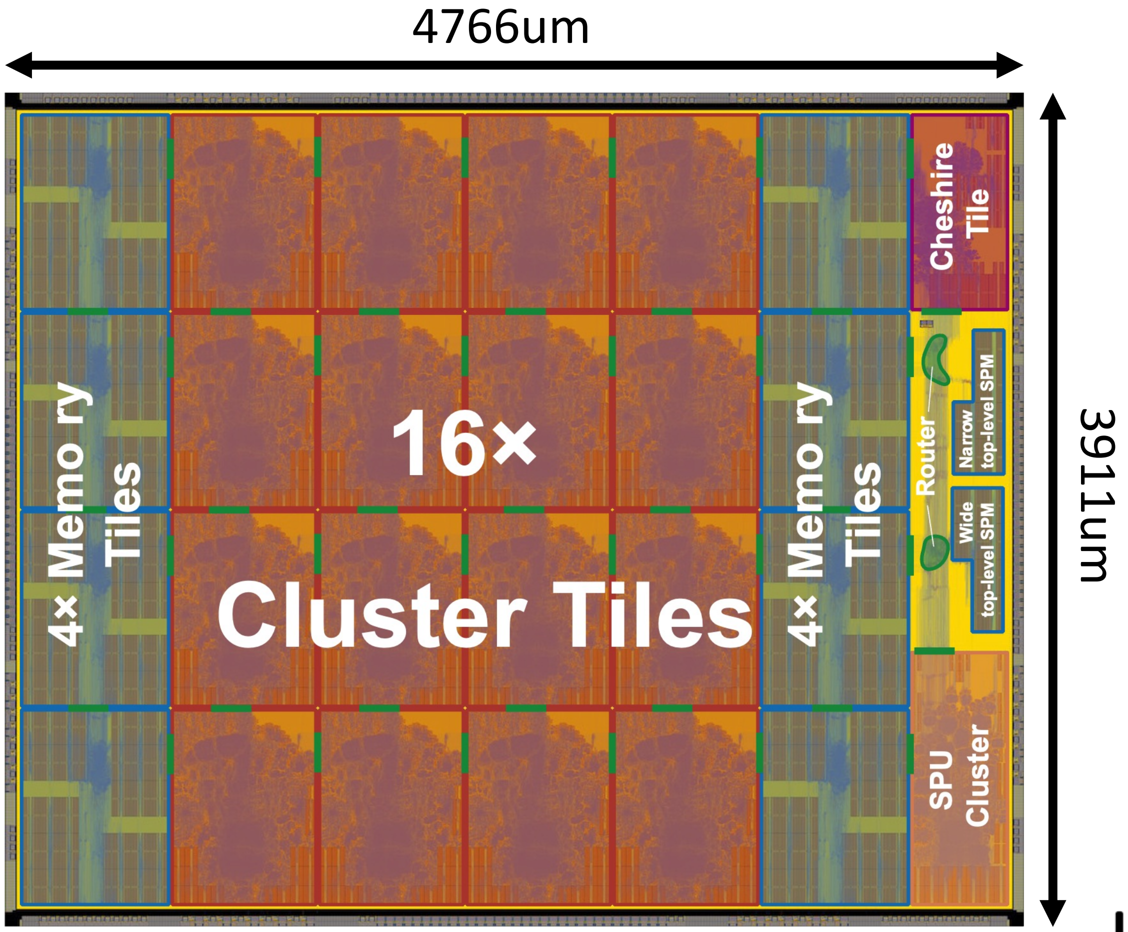
- **Tile-based System**
  - 16 Cluster Tiles
  - 8 Memory Tile
  - 1 Host core tile

- **Scaled-out with a NoC**
  - 7x4 mesh topology



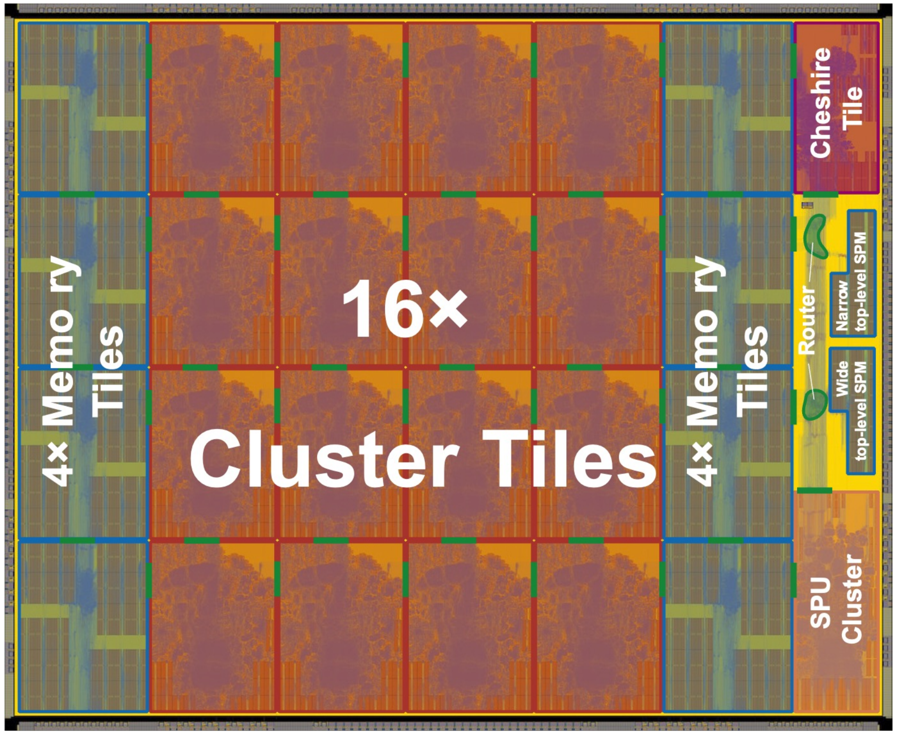
7nm Tape-out at Sept. 2025

- **Mesh topologies are great for physical design**
  - Very area efficient
  - Almost no top-level area
  - Narrow top-level channels
    - For clock, reset signals
    - Interrupt lines
    - Static tie-offs (i.e. routing information)



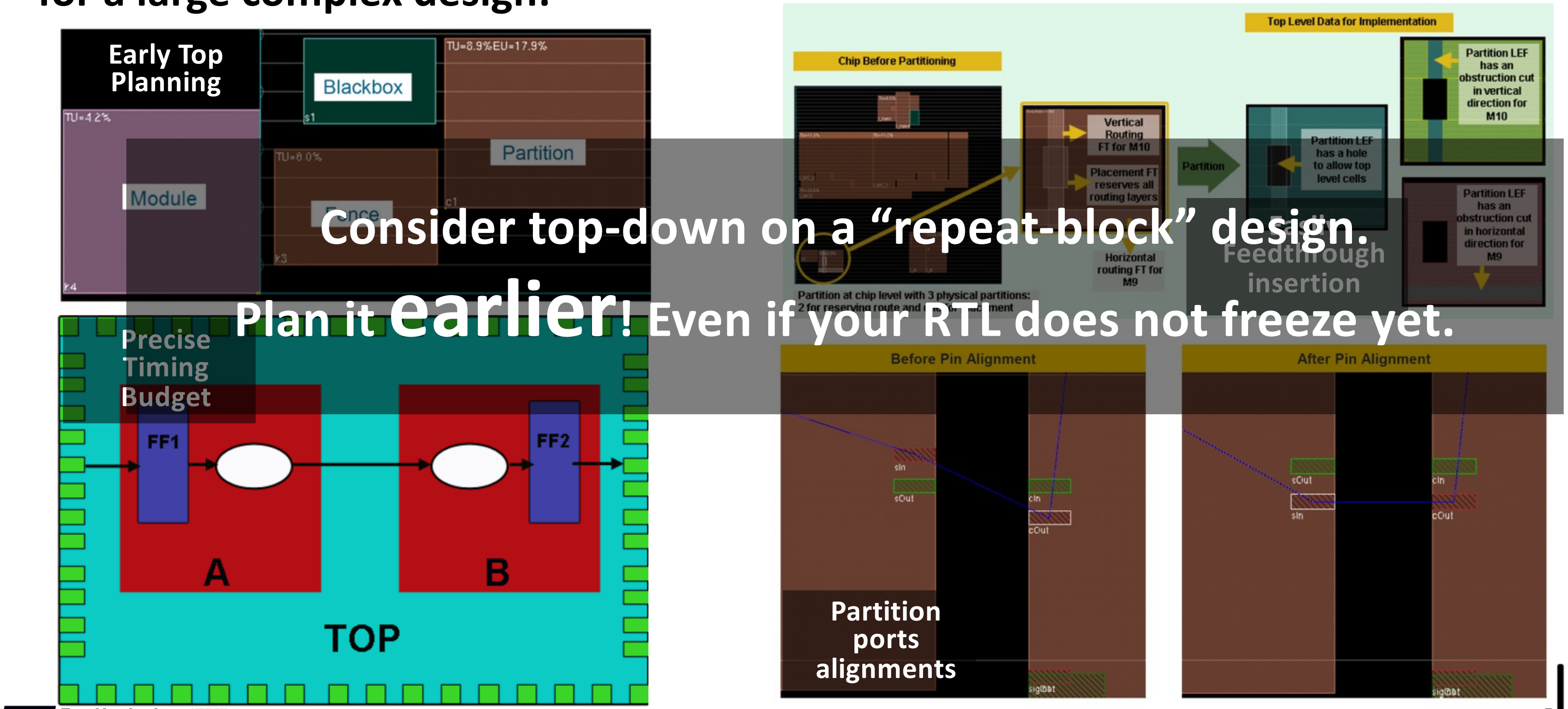
Now, new physical challenges

- **Blocks (Tiles) pin alignment**
  - Efficient routing
- **Same Tile but different physical location:**
  - Timing constraint (SDC)
  - Interface timing closing
  - Clock tree balancing
  - Cross-block DRCs (e.g. antenna) checking
  - … …



TOP-DOWN Design Planning

- **Top-down hierarchical design is more beneficial than bottom-up for a large complex design:**

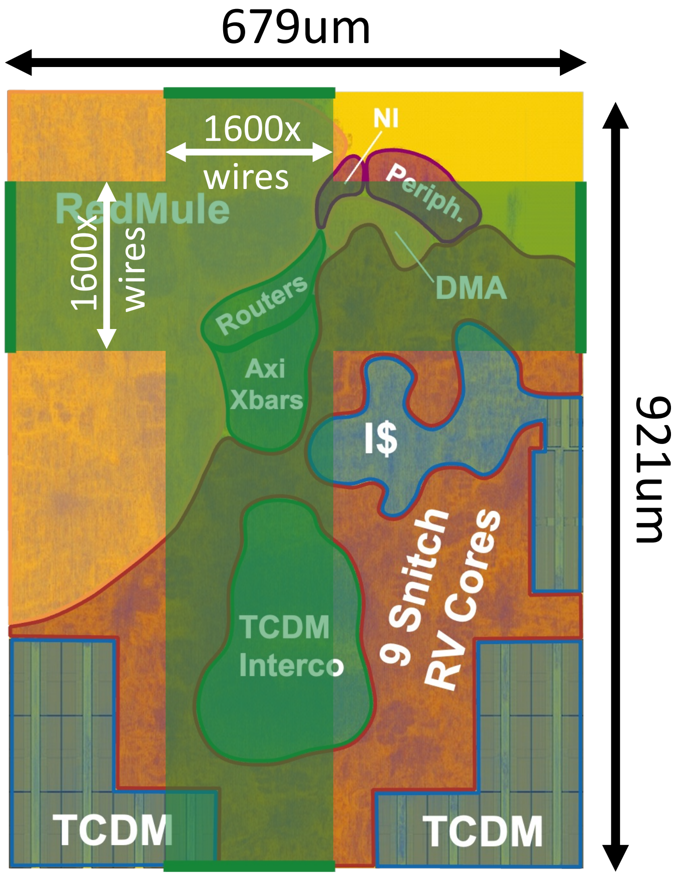

Early Top Planning
TU=4.2%
Module
Blackbox
TU=8.9%EU=17.9%
Partition
TU=8.0%
Fence
Precise Timing Budget
FF1
FF2
A
B
TOP
Top Level Data for Implementation
Chip Before Partitioning
Vertical Routing FT for M10
Placement FT reserves all routing layers
Horizontal routing FT for M9
Partition
Partition LEF has a hole to allow top level cells
Partition LEF has an obstruction cut in vertical direction for M10
Partition LEF has an obstruction cut in horizontal direction for M9
Feedthrough insertion
Partition at chip level with 3 physical partitions: 2 for reserving route and
Before Pin Alignment
After Pin Alignment
sIn
sOut
cIn
cOut
sigOut
Partition ports alignments

Consider top-down on a “repeat-block” design.

Plan it earlier! Even if your RTL does not freeze yet.

Reference: innovusUG/innovusUG23.10/Pushing_Down_a_Network_into_Block_Partitions

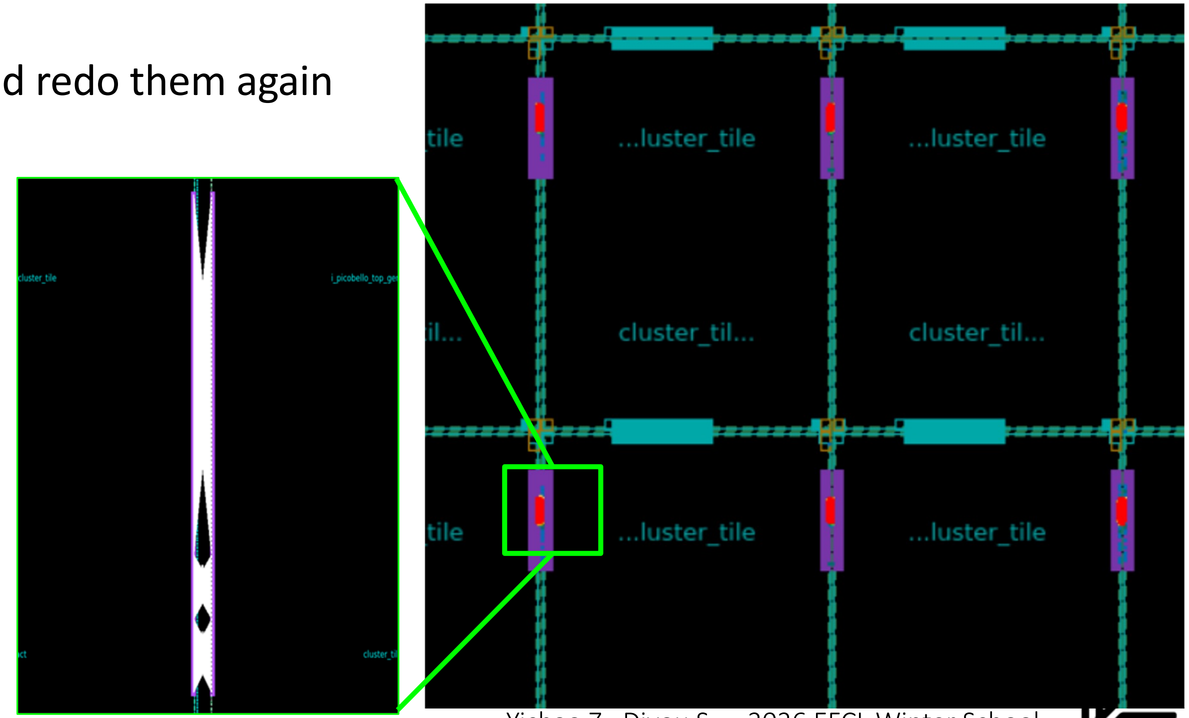
Now you have a perfect Cluster Tile, is it?



- **Dominated by compute logic**
  - RedMule & RISC-V cores
- **Only a small part for data movement**
  - NoC (Network Interface, Routers)
- **Plenty of routing resources for NoC links**
  - 2 *Mx* + 2 *My* layers for NoC routing
  - Aligned, straight, shortest (in the block view)
  - Avoid congestion
    - Using every third track

Before block partition, check is important

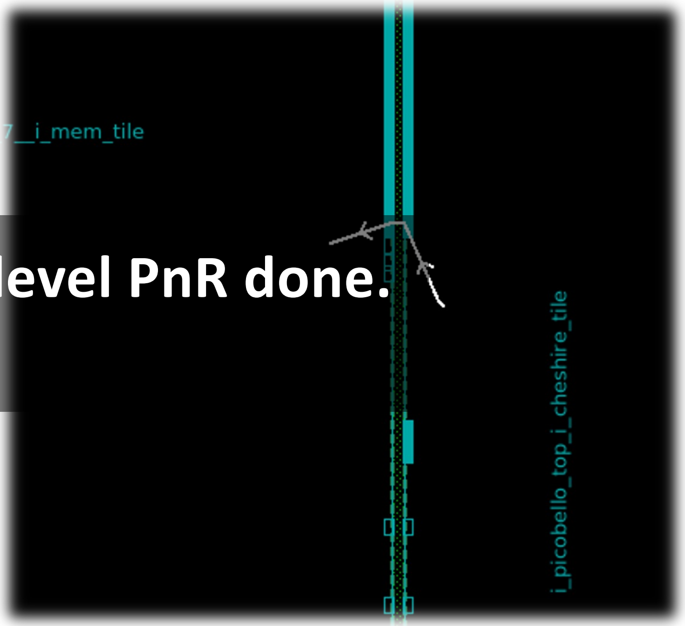
- **Small mistakes slip away, but there's no chance to fix them. A real story:**
  - TOP level design planning → partition → block PnR → Top assembling
  - Oh no, something wrong!! We need redo them again

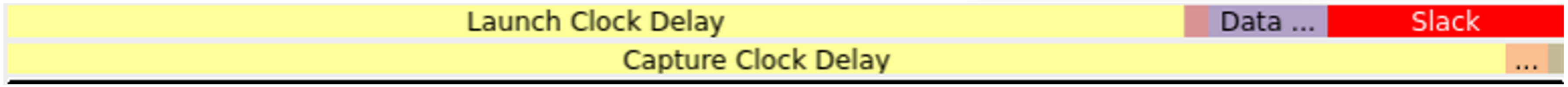


Timing signoff (an open question for you)

- **Cross-blocks paths can be only seen at the top level.**
- **For example, the hold violation:**
  - Clock skew + short path (with only few logic levels)
  - Clock skew is sensitive to hold
- **How to fix? One way:**
  - Check after top-level PnR;
  - Top release a script/PathList to block owners
  - Block owner fixes timing (and DRC)
  - Iterate it again

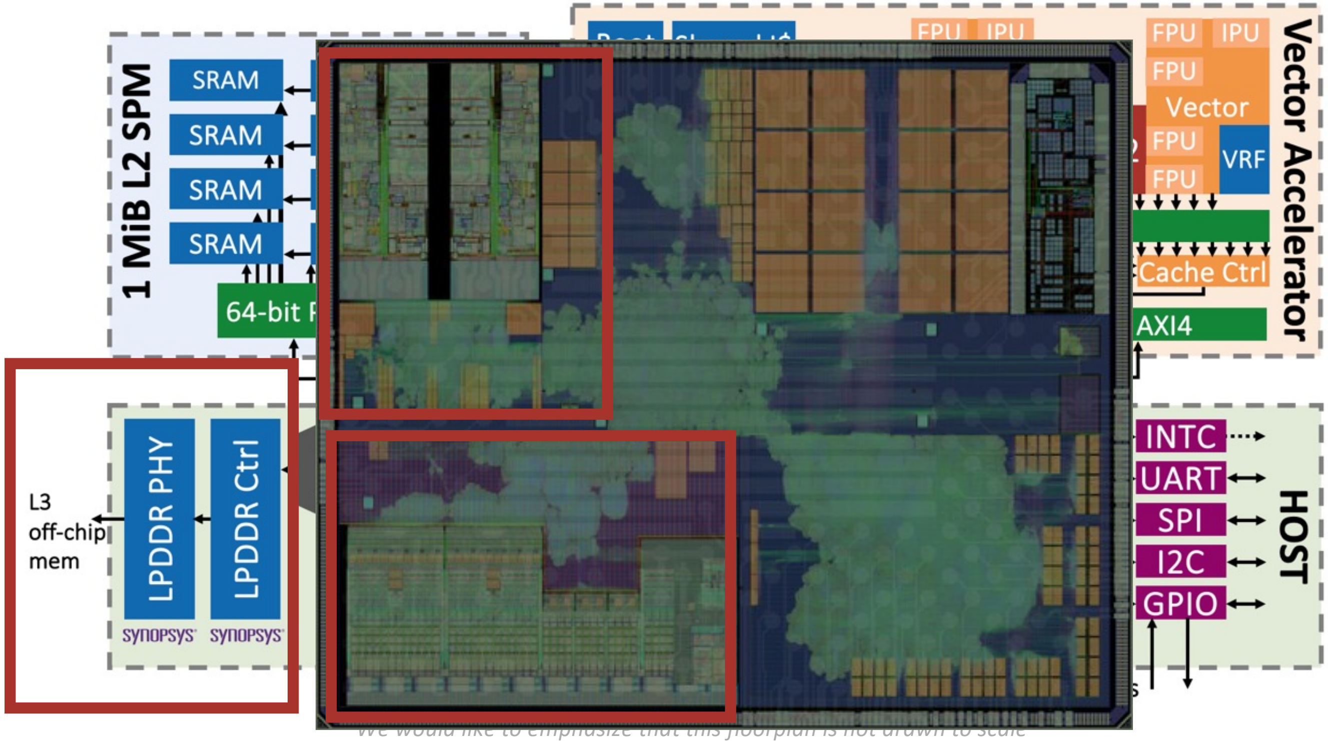
“Cross-blocks” fixing need to wait until top-level PnR done.
Any good suggestions?





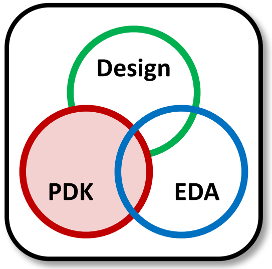
New IPs + new PDK: how to start?

- **Meet Flamingo**
  - SoC with Vector based accelerator for AI applications in GF22



GlobalFoundries™

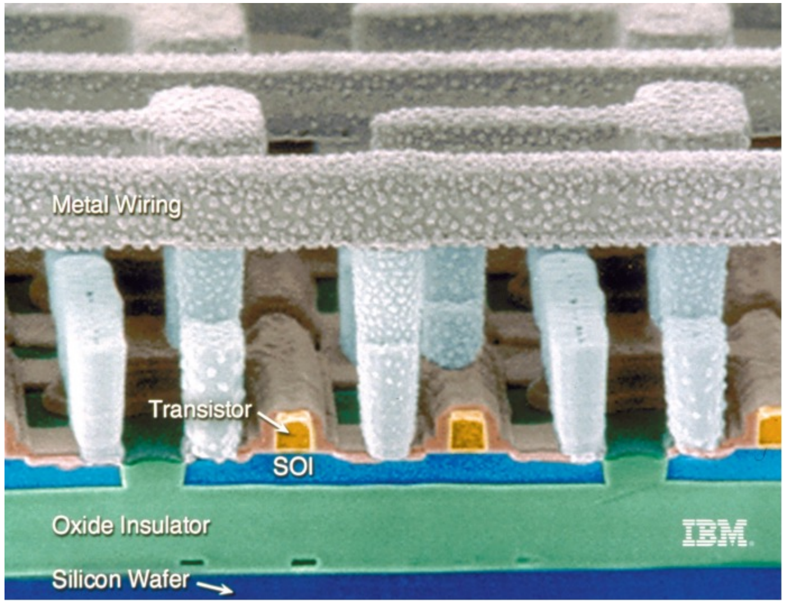
- **EDA support by** SYNOPSYS®
- **Interface IP DDR / PCIe by** SYNOPSYS®

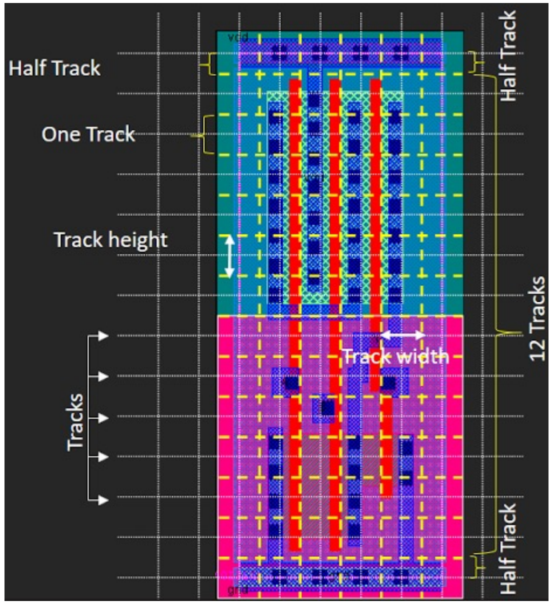


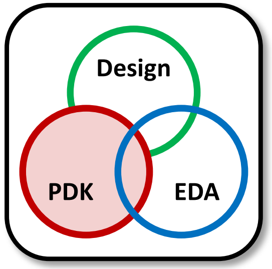
Let’s start from the PDK

- **Step 1: Understand your PDK**
  - Number and directions of metal layers
  - Standard cell track
  - Special cell/placement rules

→ **Standard Cell Guidelines**



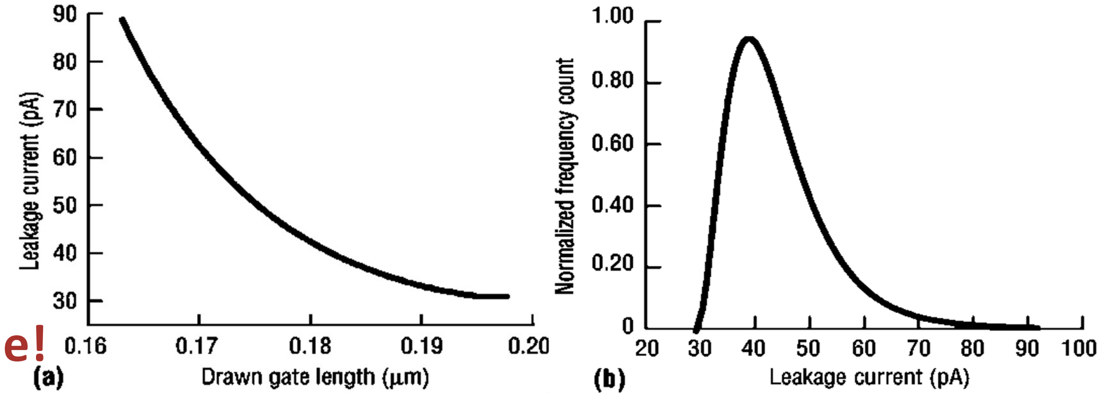


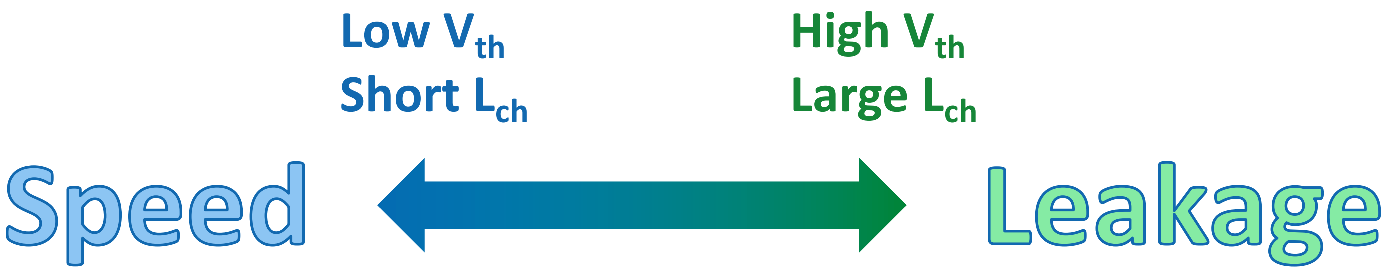


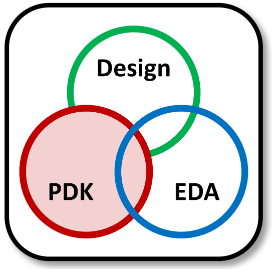
https://x.com/TubeTimeUS/status/1243950154874413057/photo/1
https://teamvlsi.com/2020/05/standard-cells-in-asic-design-standard-cells-in-vlsi.html

How to select the Std cells?

- **Step 2: Standard-Cell Library Select**
  - Dozens of available libraries in a PDK
    - Threshold Voltage
    - Channel Length
    - Design Kit
  - Can we use all of them? **No! Run Time!**
  - How should we select them?







N. S. Kim *et al.*, *IEEE Computer*, vol. 36, no. 12, pp. 68–75, 2003

How to select the Std cells?

- **Step 2: Standard-Cell Library Selection**
  - Dozens of available libraries in a PDK
    - Threshold Voltage
    - Channel Length
    - Design Kit
  - Can we use all of them?
  - How should we select them?

Design Kit

Standard logic cells
e.g. adder

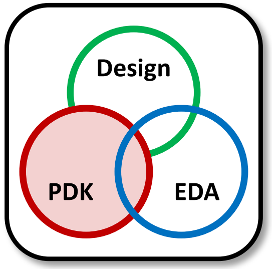
Physical cells
e.g. filler, boundary

Power cells
e.g. level shifter

Low V_{th}
Short L_{ch}

High V_{th}
Large L_{ch}

Speed ⟷ **Leakage**



How to select the Std cells?

- **Step 2: Standard-Cell Library Selection**
  - Dozens of available libraries in a PDK
    - Threshold Voltage
    - Channel Length
    - Design Kit
  - Can we use all of them?
  - How should we select them?

Example Selection:

Target: **performance**

Lib1: SL-Vth, Lmin, STD PHY

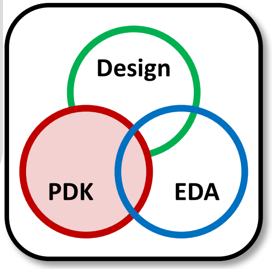
→ **Speed**

Lib2: SL-Vth, Lmid, STD PHY

→ **Speed & Hold fixing**

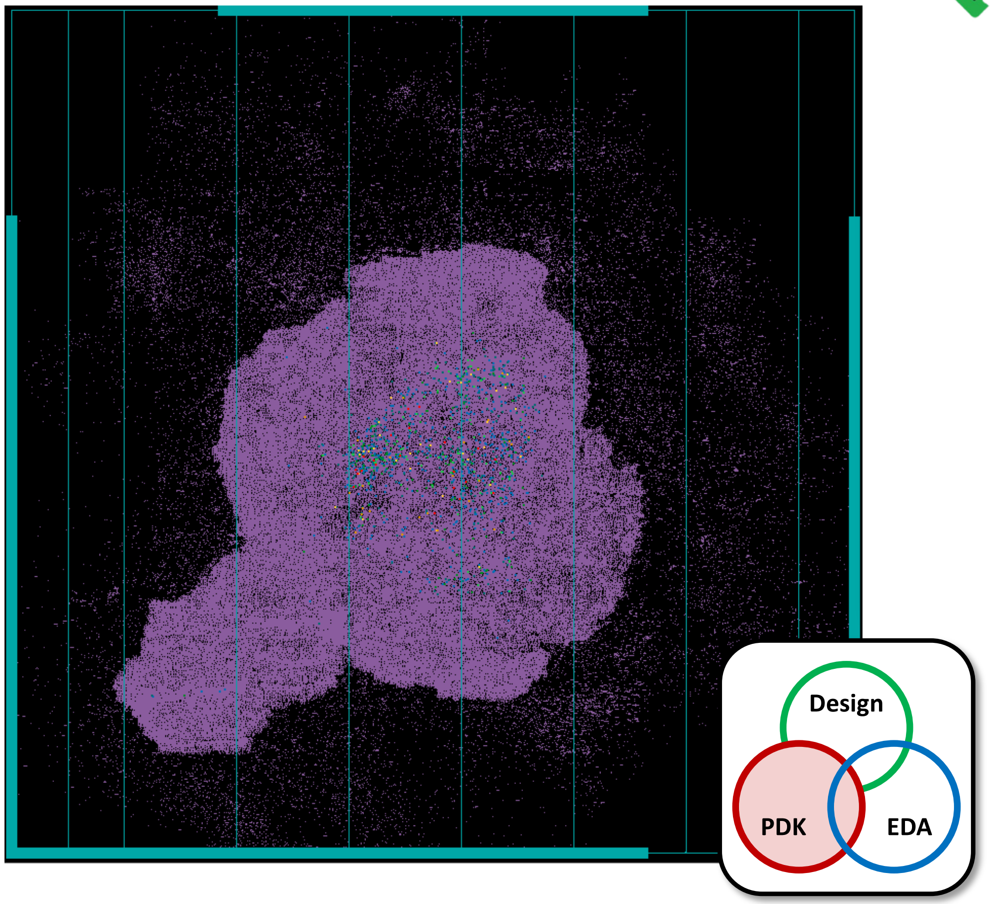
Lib3: L-Vth, Lmid, STD PHY

→ **Hold fixing and filler**



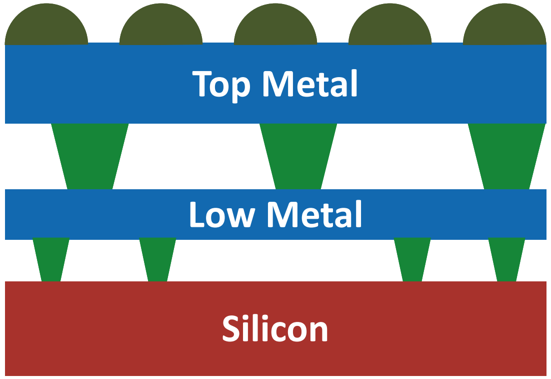
Let’s start from the PDK

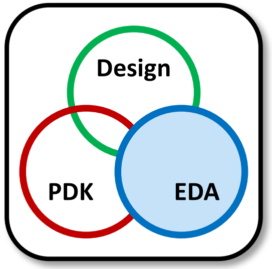
- **Step 3: Test Run**
  - There are things “hidden” in PDK
    - e.g., pin access
  - Run a small block to study
    - Timing/frequency
    - Routing
  - Can also be used to tune the basic flow!
    - Helpful when design is not frozen



Power Grid Design: Chip Level

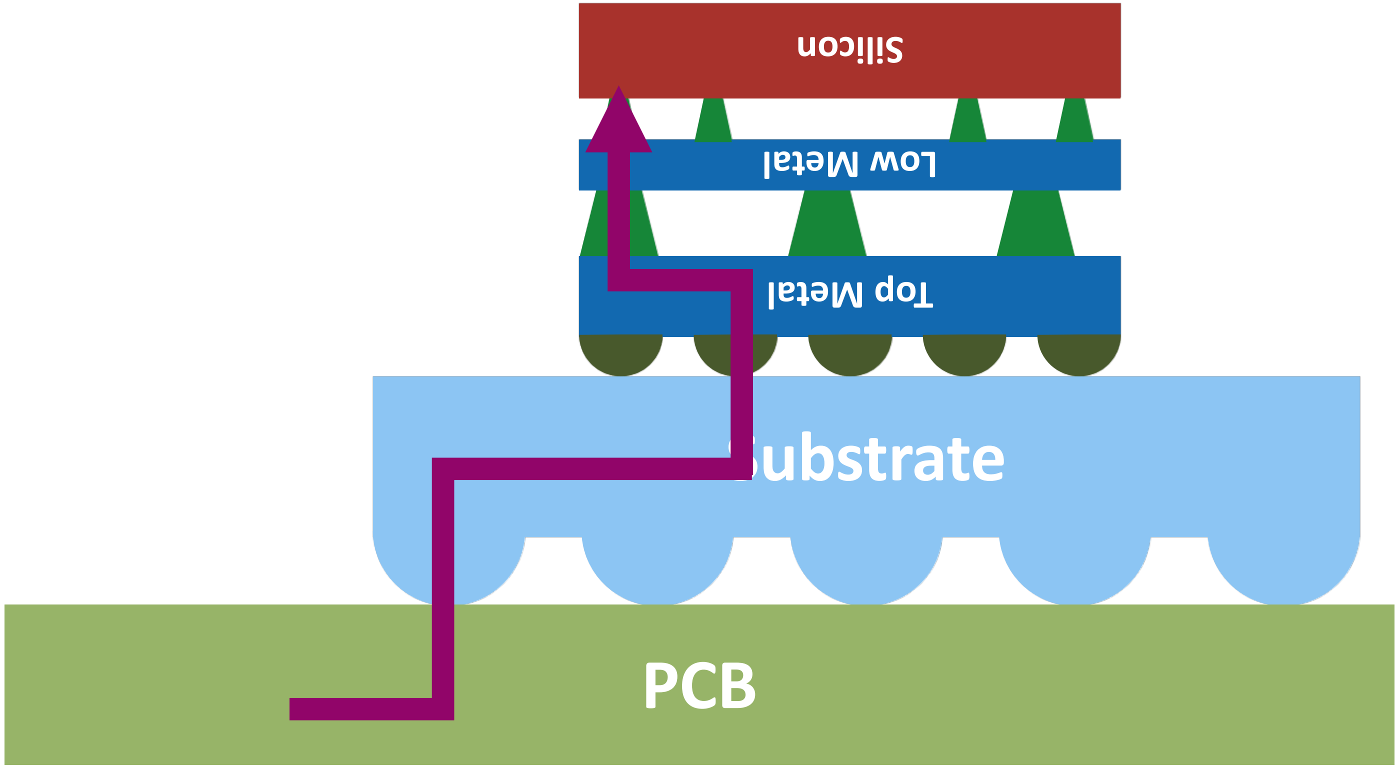
- **How does power flow into a flip-chip?**

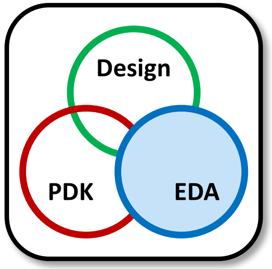




Power Grid Design: Chip Level

- **How does power flow into a flip-chip?**

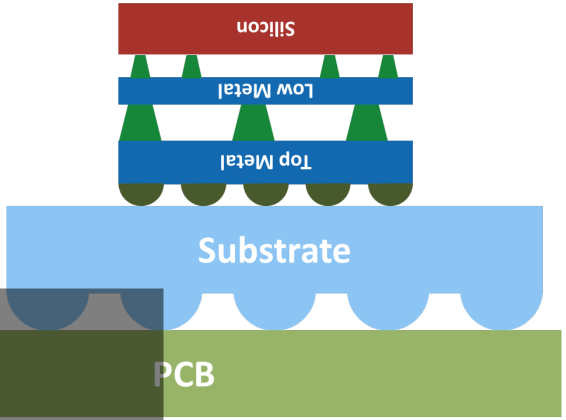


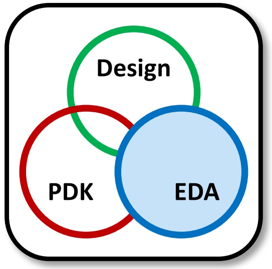


Power Grid Design: Chip Level

- **How does power flow into a flip-chip?**
  - From Silicon view: Top → Bottom
  - Need a dense mesh at top
    - Distribute the power to the chip
    - Low IR drop
- **More factors to consider on flip-chip**
  - RDL routing v.s. RDL PG mesh
  - Extra ESD cells

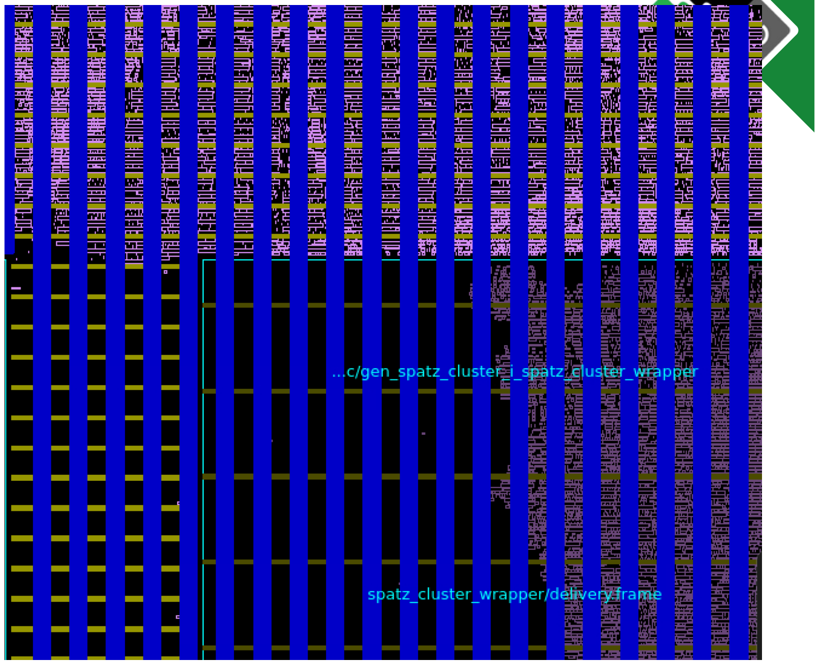
Plan in advance

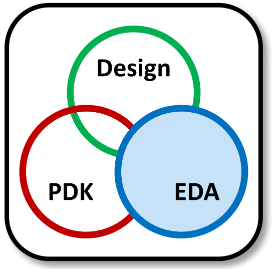




Power Grid Design: Block Level

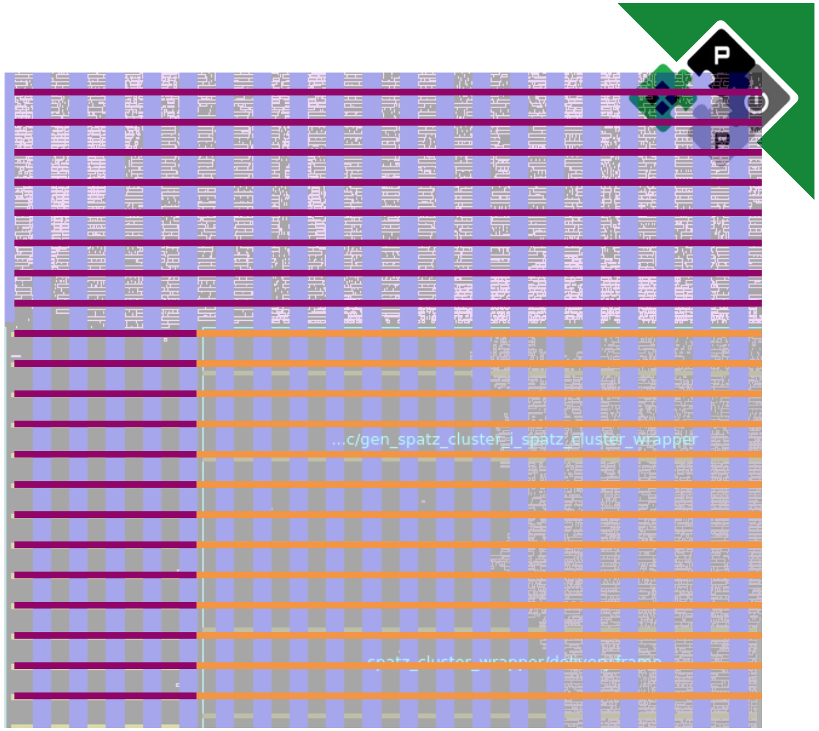
- **Key: Keep PG simple**
  - Top level integration should also be considered
  - How to connect block PG to top?
    - Aligned PG wires → Make a uniform mesh on the whole chip
      Ideal for IR drop and uniformity

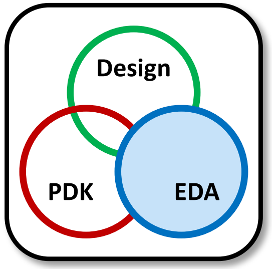

...c/gen_spatz_cluster_i_spatz_cluster_wrapper
spatz_cluster_wrapper/delivery frame



Power Grid Design: Block Level

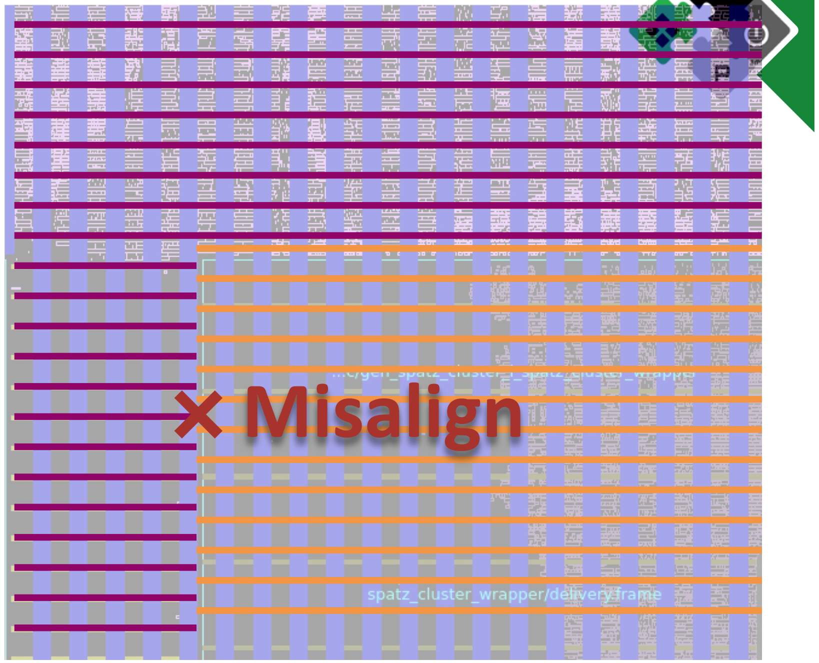
- **Key: Keep PG simple**
  - Top level integration should also be considered
  - How to connect block PG to top?
    - Aligned PG wires → Make a uniform mesh on the whole chip **Ideal for IR drop and uniformity**

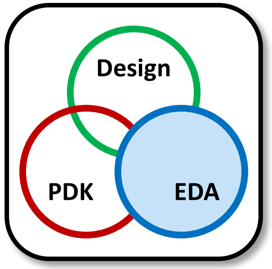




Power Grid Design: Block Level

- **Key: Keep PG simple**
  - Top level integration should also be considered
  - How to connect block PG to top?
    - Aligned PG wires → Make a uniform mesh on the whole chip
      Ideal for IR drop and uniformity
      But what will about a last-minute floorplan eco?

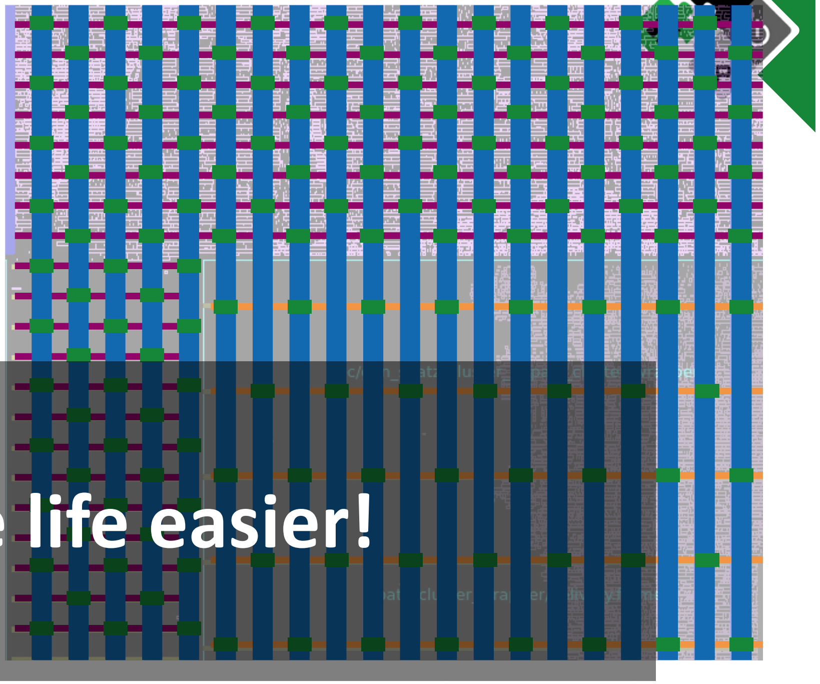

...c/gen_spatz_cluster_1_spatz_cluster_wrapper
✕ Misalign
spatz_cluster_wrapper/delivery.frame

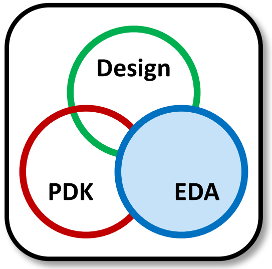


Power Grid Design: Block Level

- **Key: Keep PG simple**
  - Top level integration should also be considered
  - How to connect block PG to top?
    - Aligned PG wires → Make a uniform mesh on the whole chip
      Ideal for IR drop and uniformity
      But what will about a last-minute floorplan eco?
    - Through Vias → Separate the low metal mesh
      Power delivers **from top**!
      More freedom when an eco is needed

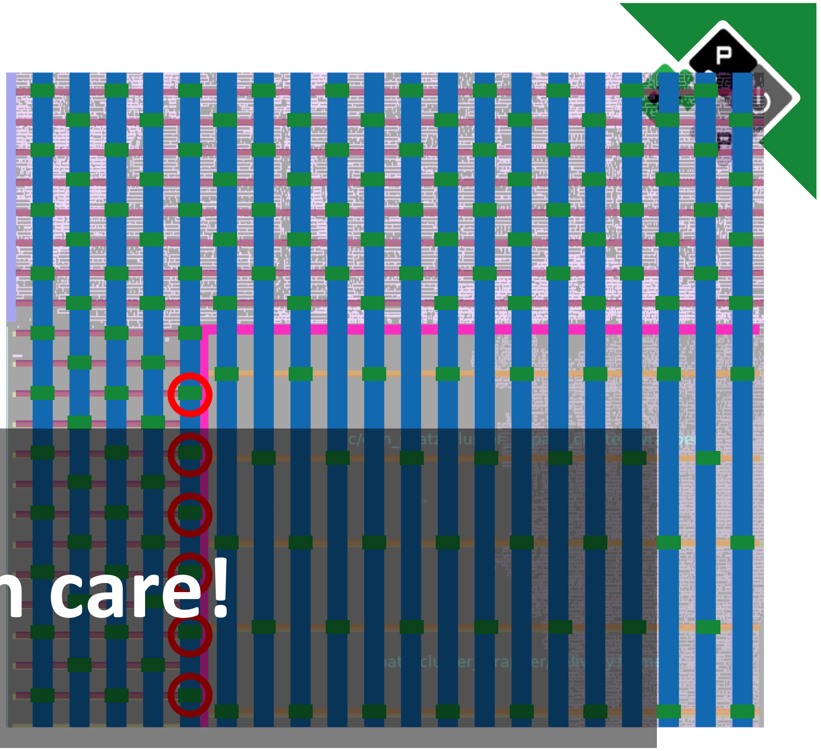
Keep PG simple, make life easier!



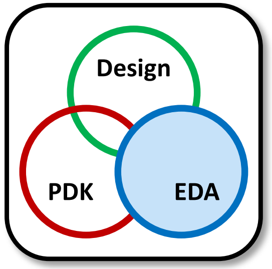


Power Grid Design: Block Level

- **Power Ring**
  - Needed at chip-level for wire-bounding → Distribute power
  - Sometimes useful to connect two different PG mesh pattern
  - But does **NOT** help in power delivery
  - Easy to create PG DRCs

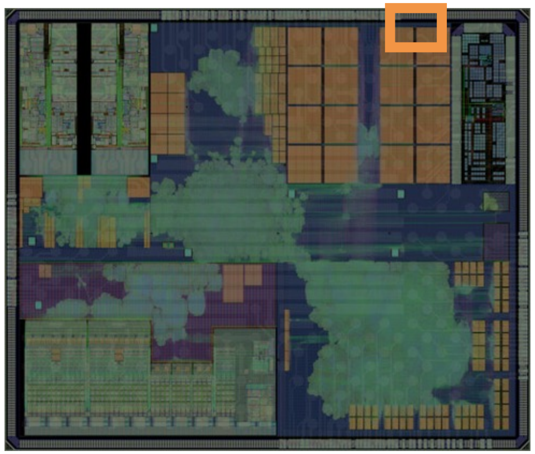


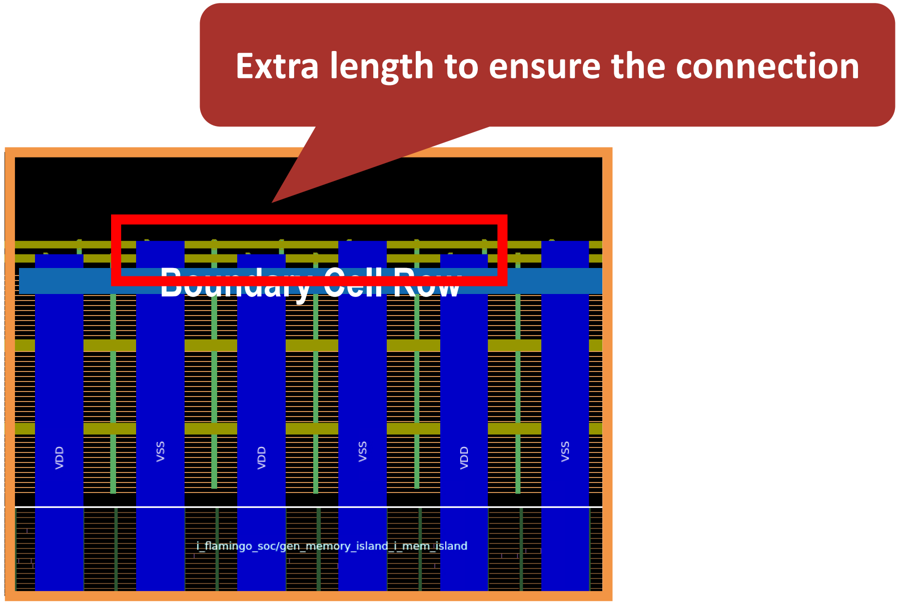
Use PG ring with care!

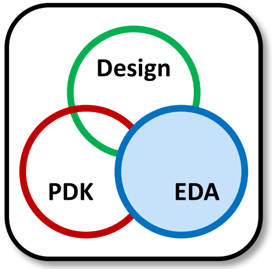


Power Grid Design: Top Level

- **Pitfalls #1: Edge of the design**
  - Sometimes needed for pad connection
  - Boundary cell power supply → slightly longer PG mesh

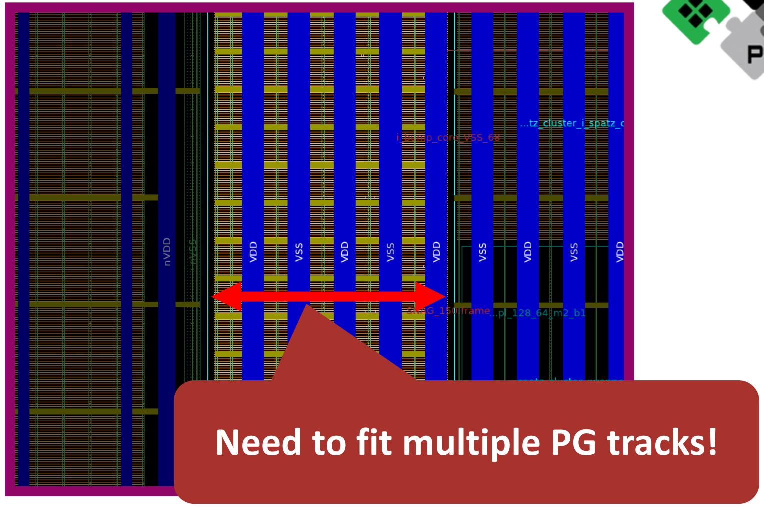



Extra length to ensure the connection
Boundary Cell Row
VDD
VSS
VDD
VSS
VDD
VSS
i_flamingo_soc/gen_memory_island_i_mem_island

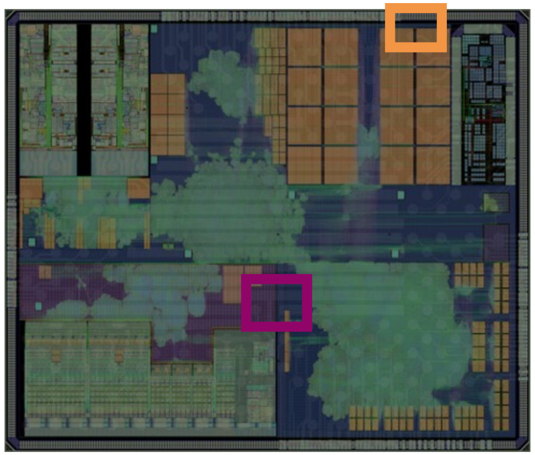


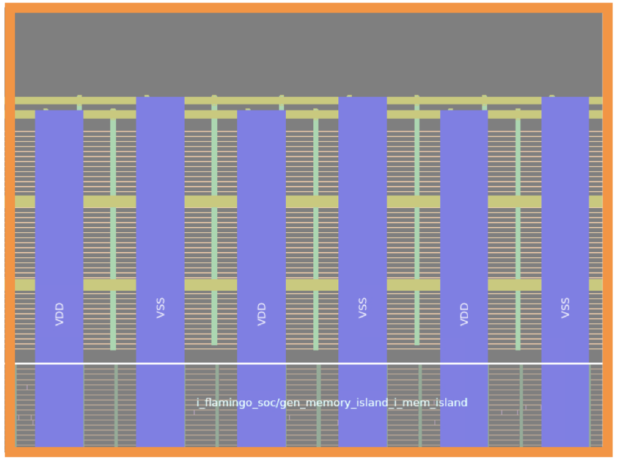
Power Grid Design: Top Level

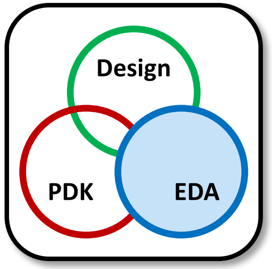
- **Pitfalls #1: Edge of the design**
  - Sometimes needed for pad connection
  - Boundary cell power supply → slightly longer PG mesh
- **Pitfalls #2: Channel between the blocks**
  - Broken mesh
  - Connectivity may still good → Difficult to find



Need to fit multiple PG tracks!

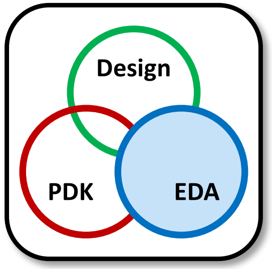






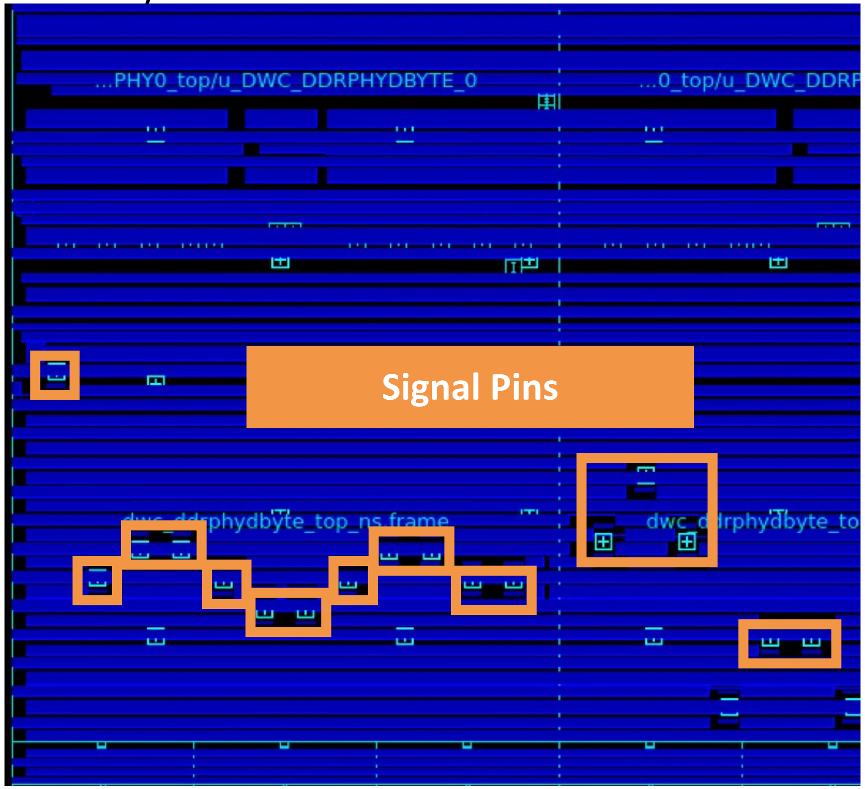
Power Grid Design: 3rd-Party IP

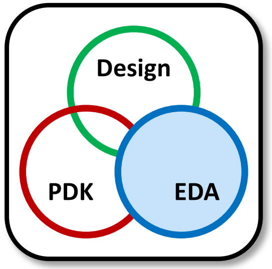
- **Special PG Requirements at IP Level**
  - Minimum density
  - Multiple voltage regions
  - Special PG directions
  - But don’t forget how the power is delivered!
    - PG will not work if we cannot route power down



Power Grid Design: 3rd-Party IP

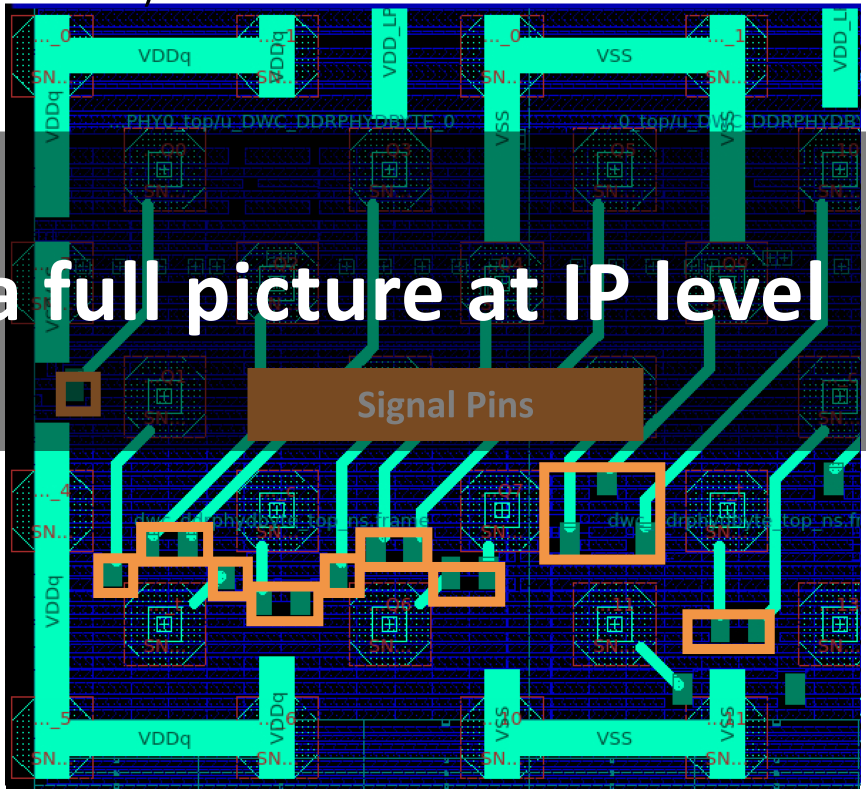
- **Example: LPDDR in Flamingo**
  - Dense PG at OI layer (top metal below LB)
    - Satisfied density > 70%



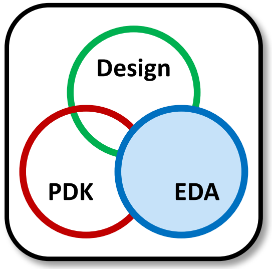


Power Grid Design: 3rd-Party IP

- **Example: LPDDR in Flamingo**
  - Dense PG at OI layer (top metal below LB)
    - Satisfied density > 70%
  - **Costs**
    - Cannot connect to top mesh
    - Violate metal density rule (< 80%)
    - Did not reserve space for sign
    - Opposite PG direction

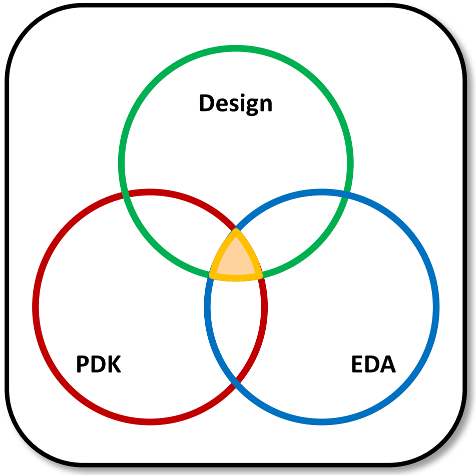


Think a full picture at IP level



Today’s takeaway

- **Physical design is not “push-button”**
  - Understand your design (data flow, wiring complexity, etc.)
  - Understand the PDK (lib pack, routing resources, package, etc.)
  - Understand the EDA (Tool is just “tool”, but you are the boss)



- **We suggest:**
  - Do a physically feasible design
  - Do RTL signoff before you go to physical stage
  - Select reasonable design kit (PG pattern, lib pkg, 3rd-party IPs … …)

- **Make your chip shine**

PULP Platform
Open Source Hardware, the way it should be!

Q&A

Yichao Zhang yiczhang@iis.ee.ethz.ch

Diyou Shen dishen@iis.ee.ethz.ch

Institut für Integrierte Systeme – ETH Zürich
Gloriastrasse 35
Zürich, Switzerland

DEI – Universitá di Bologna
Viale del Risorgimento 2
Bologna, Italy

ETH zürich ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

@pulp_platform
pulp-platform.org
youtube.com/pulp_platform