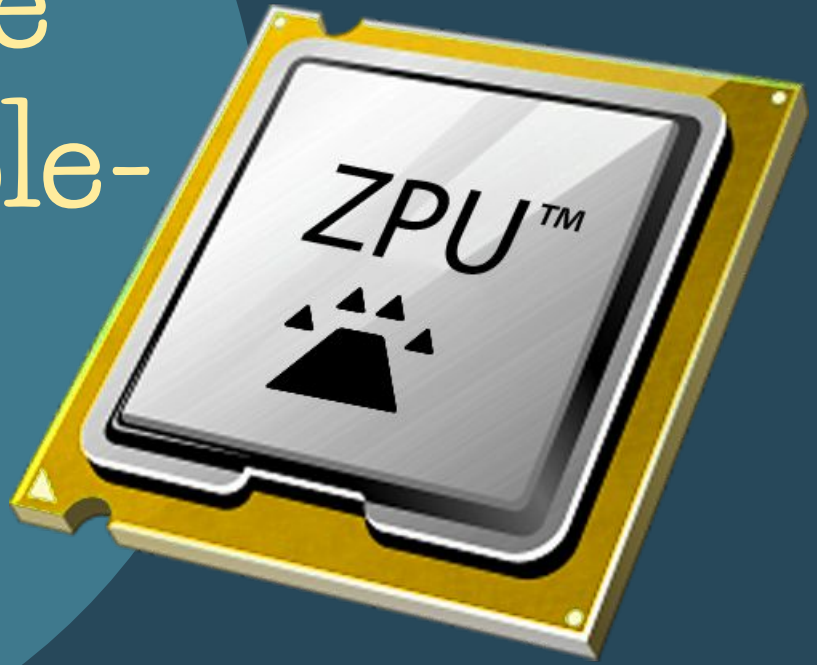
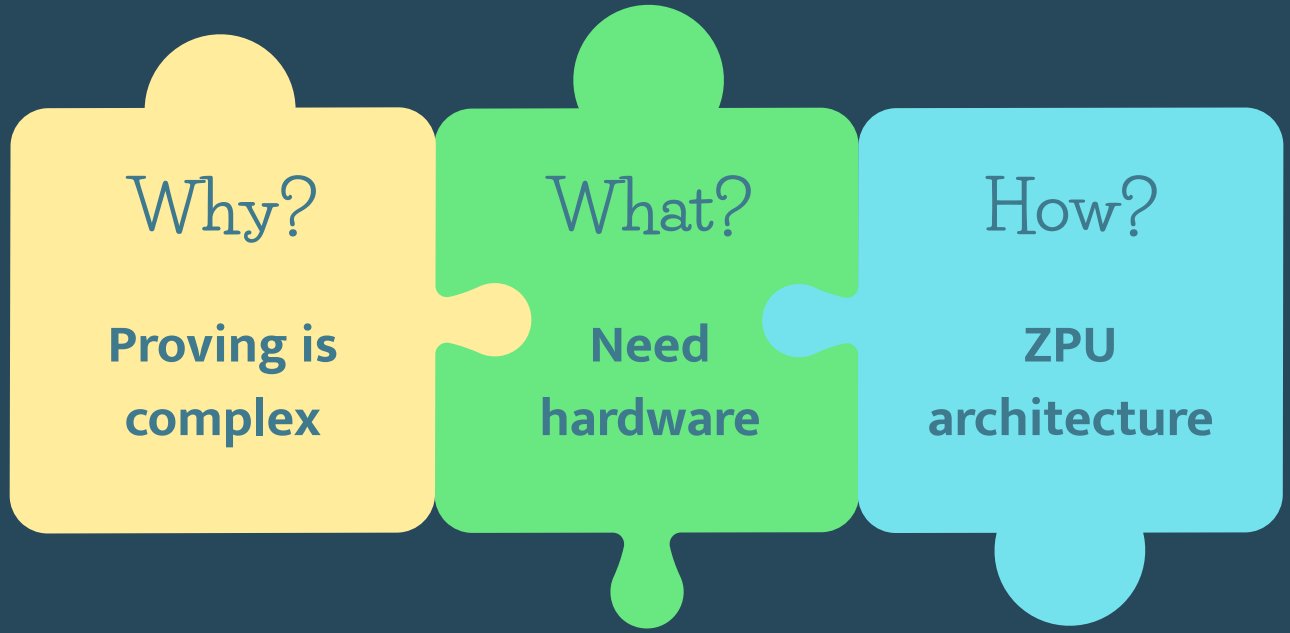
ZPU: The Hardware Path to Verifiable-Everything

Yuval Domb
Chief Architect, Ingonyama



Why are we here?



Content may contain the speaker's opinion

The holy grail of

ZKP

Medical Record Sharing

Anonymous Identification

Client-Side
Privacy-Centric

Anonymous Positioning

Financial Record Sharing

Blockchain Bridges

Storage As a Service

Server-Side
Verification-Centric

Blockchain Rollups

Computation As a Service

AI Inference As a Service

The holy grail of

Server-Side ZKP

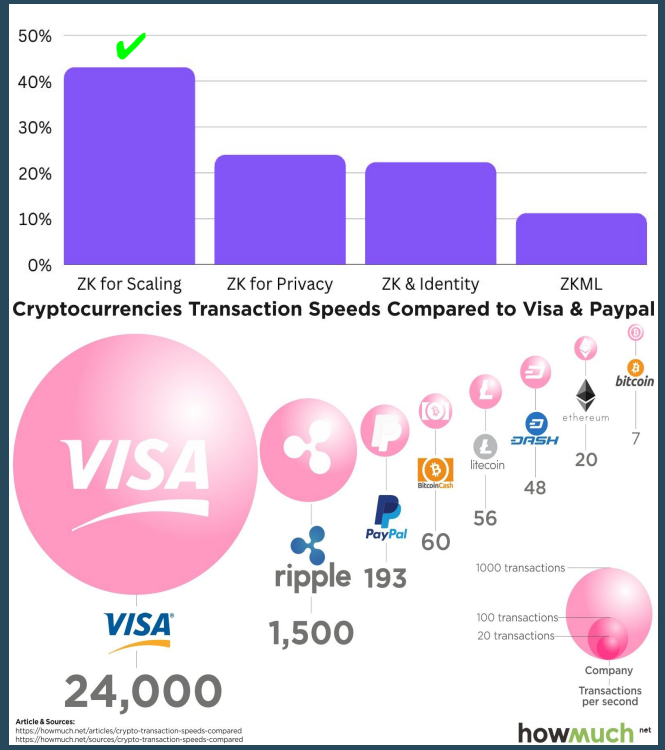
Verifiable-Everything

The problem

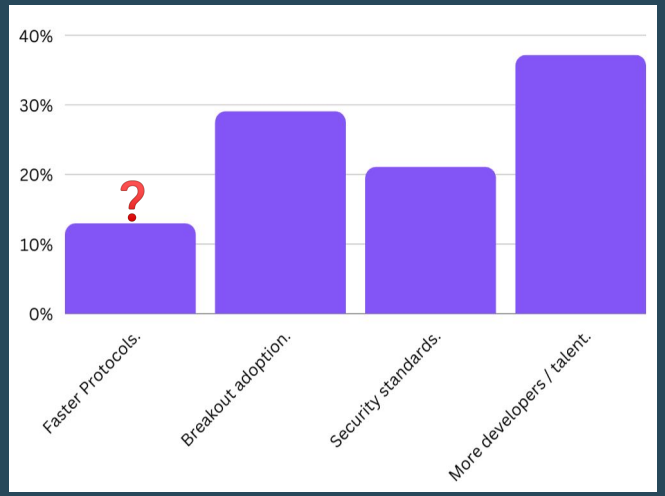
- Proving something is significantly *harder* than what's being proved
  - Arithmetization blow up
  - Polynomial manipulation - interpolation, convolution, NTT
  - Merkle trees and hashes
  - KZG and EC
  - Large finite-field arithmetics

Common perception: no problem!?

ZK highest growth?



ZK need most to succeed?



Source: www.zkvalidator.com, The State of ZK, Q2 2023.
Based on a series of polls from the @zeroknowledgefm account on Twitter.

Is there a problem?

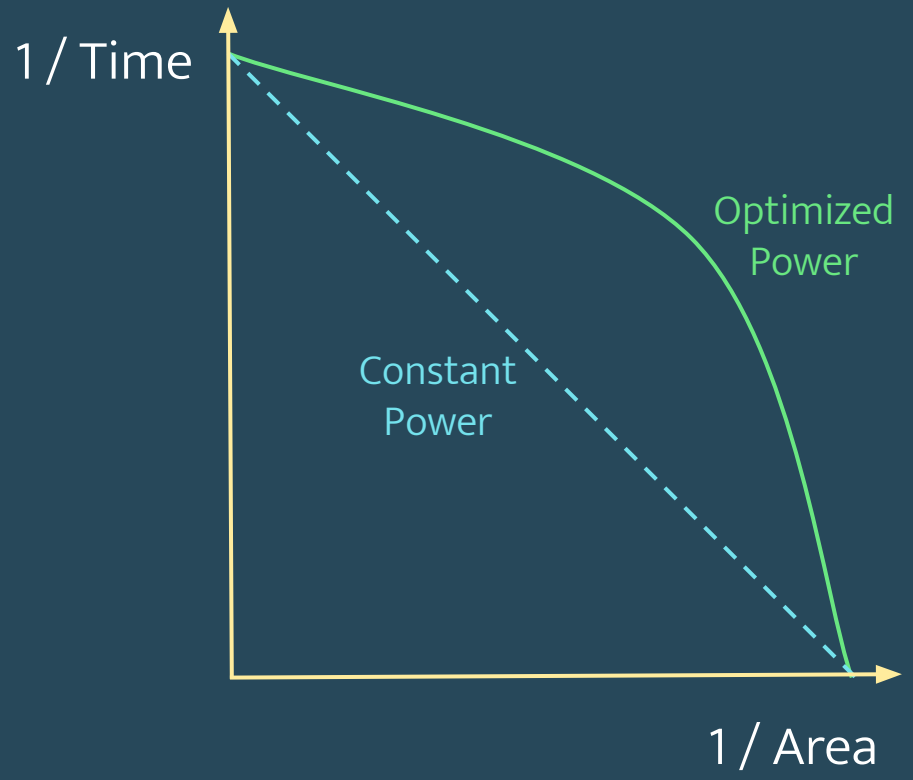
- Example: ZK-EVM for Ethereum scaling
  - Proves ZK-rollup state evolution
  - Delays finality
  - Affects costs
- **If cost of proving was graceful wouldn't everything be verifiable?**

The framework: how to measure?

- Cost: Power, Time, Area
- Performance: Security (soundness), Zero knowledge, Quantum security
- Work: Size of problem

$$\frac{\text{Power}/\text{Security}}{\text{Work}}$$

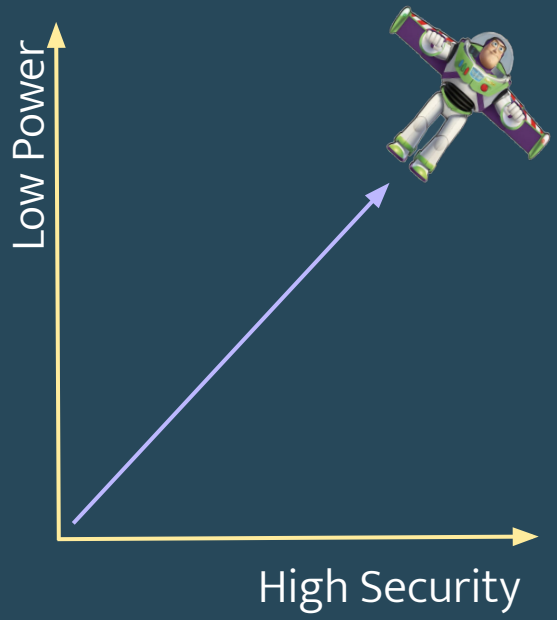
Why power?



ZK-rollup: not yet good enough

	ETH	ZK-RU	ZK-RU w/ aggregation
ETH gas/tx	High	Med	Low
TPS	Low	Med	High
RU gas/tx	None	High	High

prover power too high

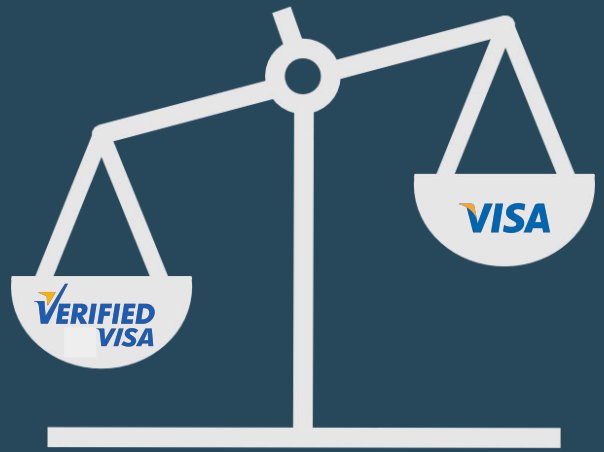


Who took our power?

	Arithmetic	Compute	Input interface	Output interface	Constant memory	Dataflow complexity
MSM	EC	O(n)	High	Low	High (setup)	Med
NTT	FF	O(nlogn)	High	High	High (twiddle)	High
Hash	FF	$O(n^2)$	High	Low	High	Med

Where is our effort spent? (misconceptions)

- Abstraction is king
  - Top-down methodology isn't power efficient
  - The developer isn't the client
  - Open-source encourages over-abstraction
  - Evolved abstraction layers hide complexity well
- Security sells
  - Nice-to-have feature
  - Maybe in specialized areas



How to solve?

Minimize: Power / Security

Engineering

Mathematics

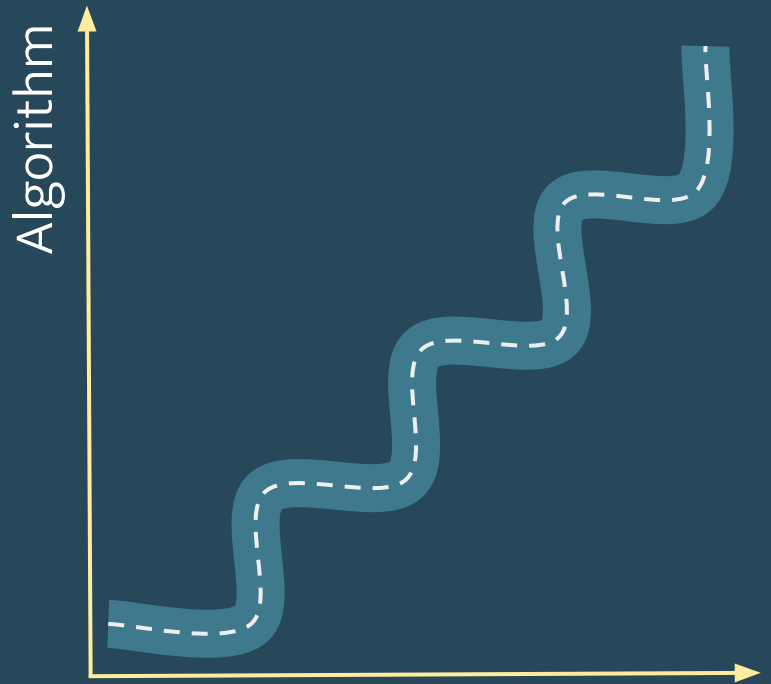
Hardware

Algorithm

The hardware evolution

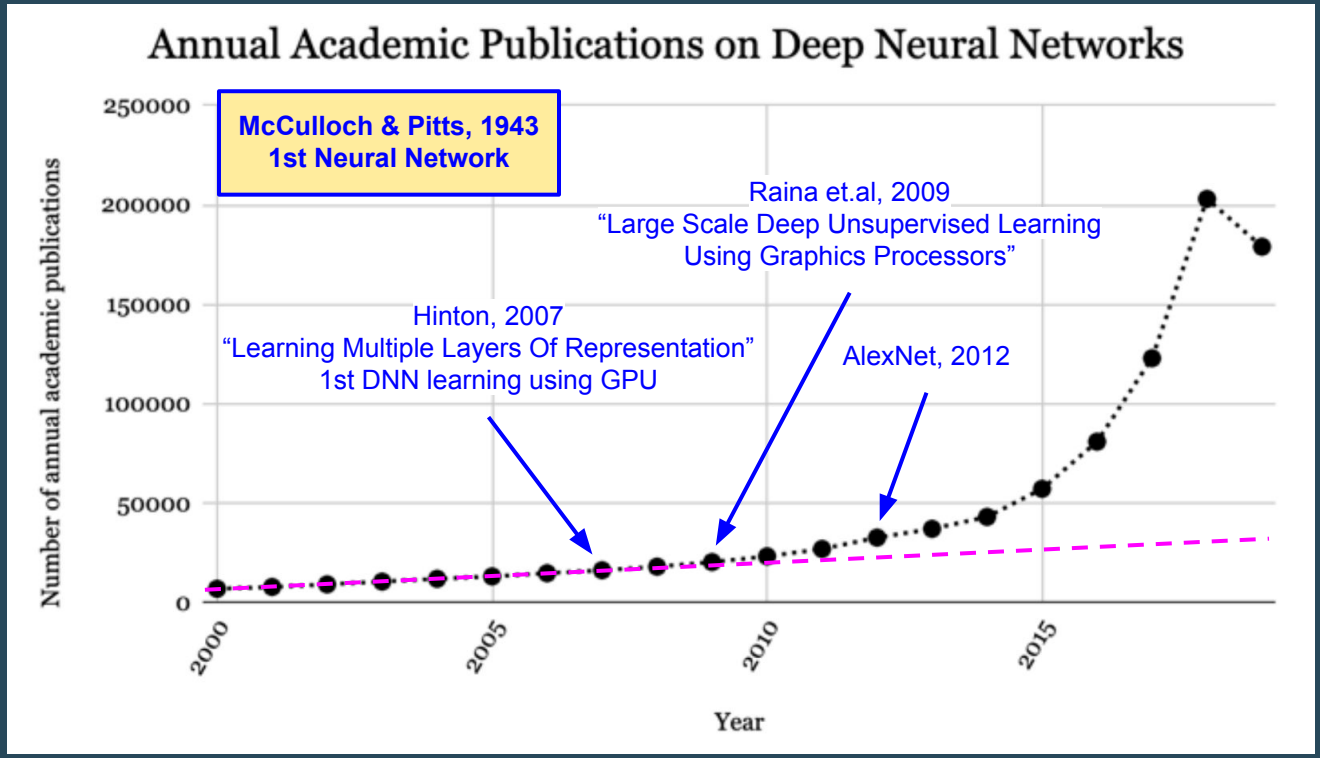
Available hw induces new algorithms

Algorithmic needs push for new hw



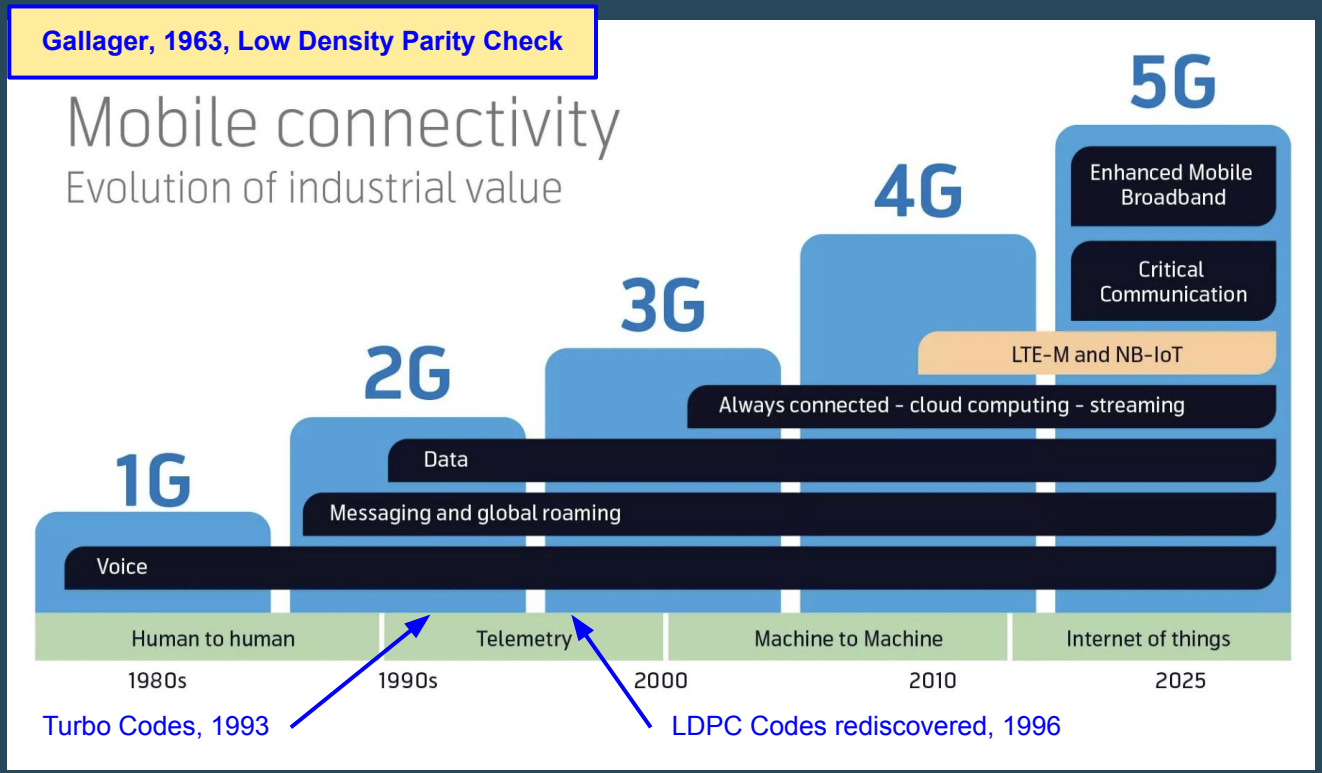
The thinking outside-the-box process

Hardware evolution: DNN winter 1943-2007



Sources: 1) Keith D. Foote, 2022, A Brief History of Deep Learning, 2) Reid Pryzant, A Quick Tour of Deep Learning History, 3) Wilhelm Klat et.al, 2022, A Blueprint for Computer Vision Testing in the Circular Economy

Hardware evolution: LDPC winter 1963-1996



Sources: 1) Wikipedia, Low-density parity-check code, 2) Matthew Clark-Massera, 2023, Connectivity - LTE-M vs NB-IoT, Coverage, Providers and Roaming

Available hardware platforms

	CPU 	GPU 	FPGA 
Thread parallelism	Low	High	High
Scheduling	OS	RTOS	Deterministic
Memory scheme	Cache-based	Mixed	Deterministic
Power/security	High	High	High
Arithmetic	FP64	FP32	Fixed point
Programming	Easy	Med	Hard

Available hardware platforms

	CPU	GPU	FPGA
			
Thread parallelism	Low	High	High
Scheduling	OS	RTOS	Deterministic
Memory scheme	Cache-based	Mixed	Deterministic
Power/security	High	High	High
Arithmetic	FP64	FP32	Fixed point
Programming	Easy	Med	Hard

Future ZKP hardware not-to-do's

- Grid, e.g. GPU farms
  - ✗ Doesn't improve power/security
- Very flexible ZKP ASIC
  - ✗ Doesn't improve power/security much
- Very customized ZKP ASIC
  - ✗ Doesn't extend beyond customized operating point
  - ☐ May be suited for client-side or for a future standard

The ZKP hardware landscape

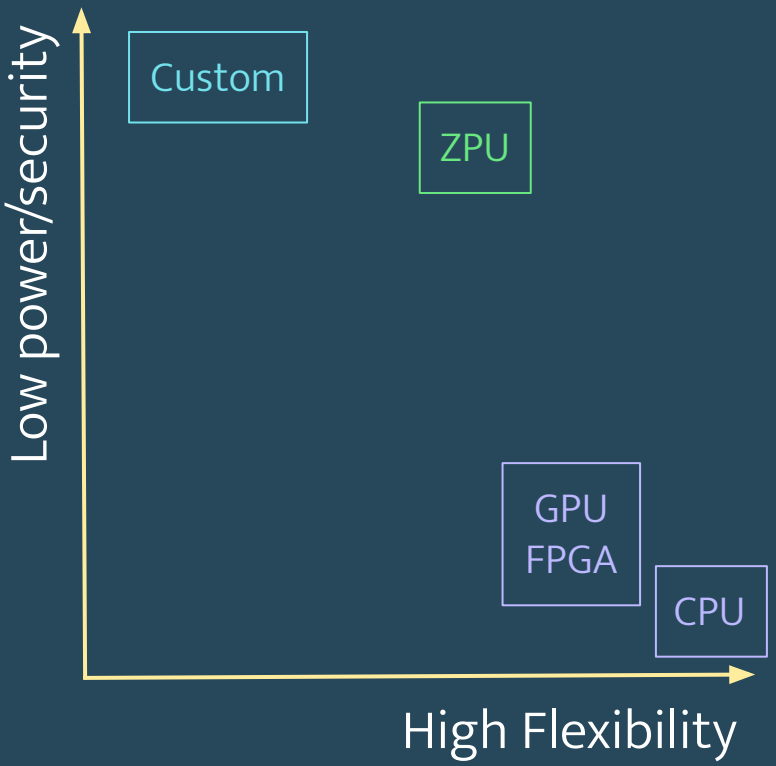
Very flexible

ZPU

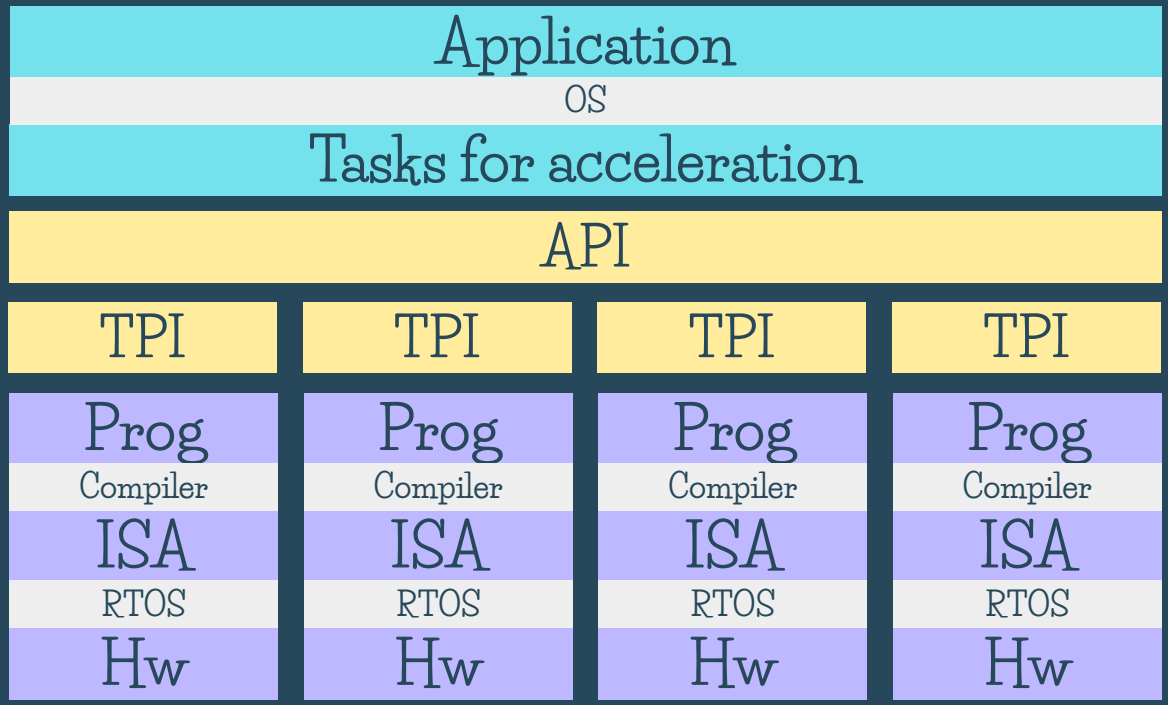
Managing Exploration vs. Exploitation

Very customized

Power/security vs. flexibility



The ZKP ecosystem



Ingonyama's ecosystem

Application

OS

Tasks for acceleration

Unified API

ICICLE

BLAZE

ZPU-TPI

CPU

GPU

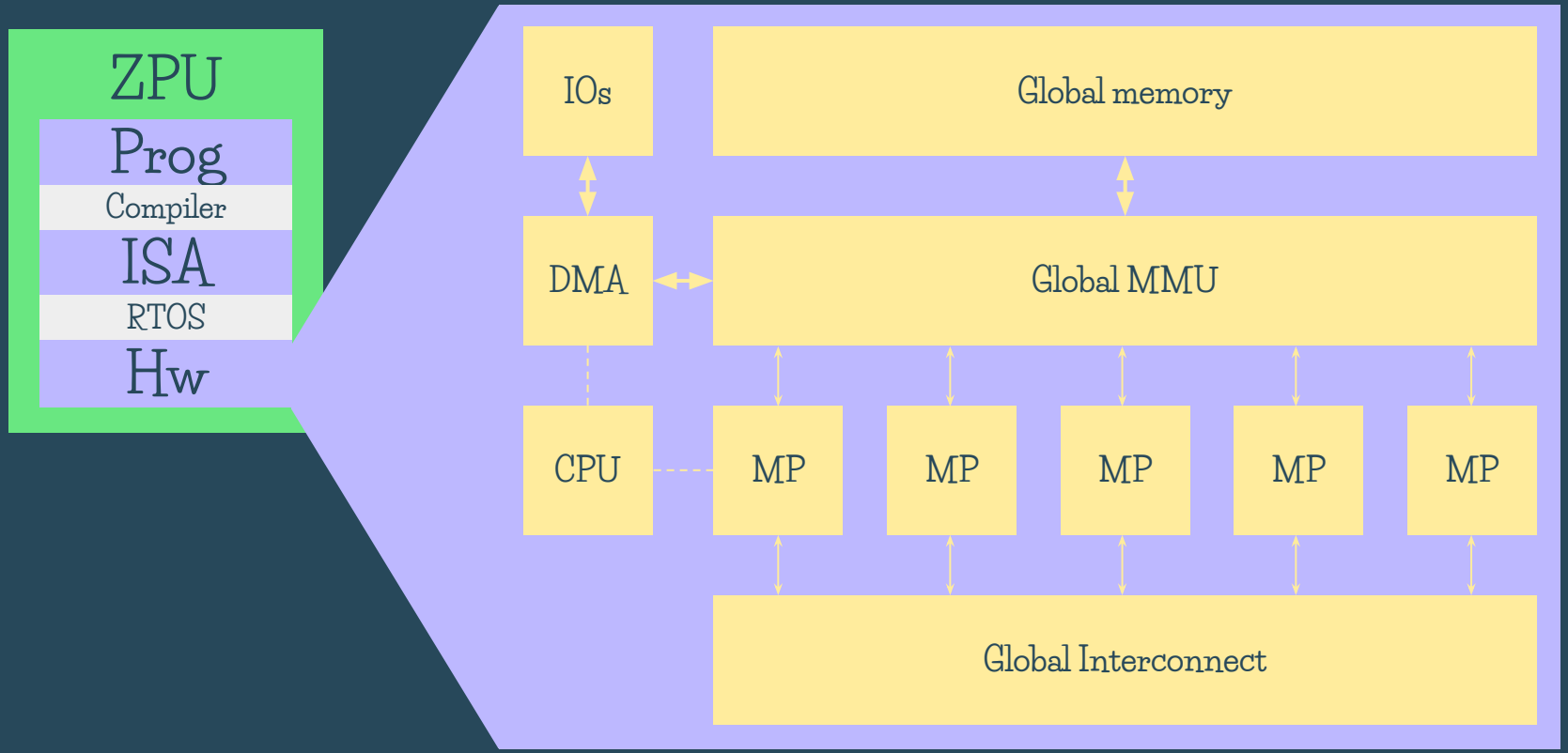
FPGA

ZPU

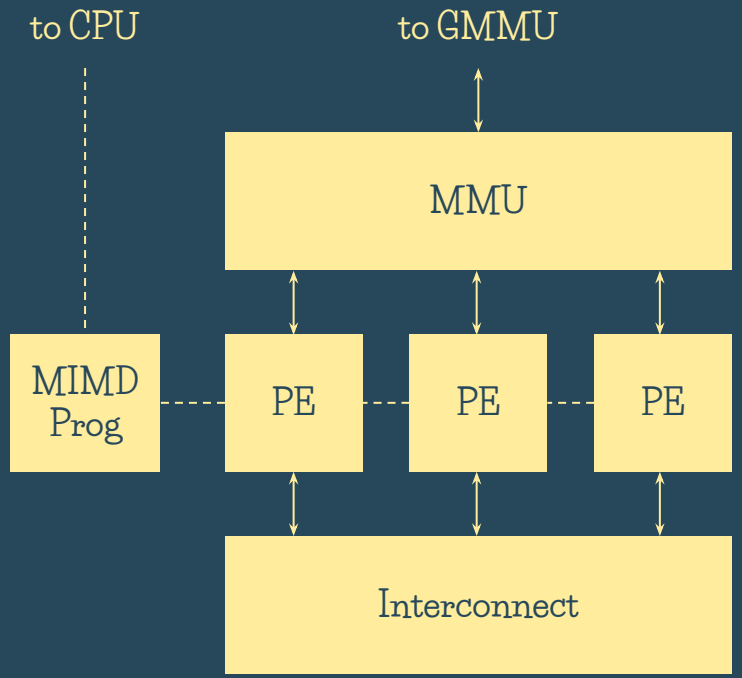
ZPU architecture: what's important?

- Large-integer modular arithmetic supported natively
- CT butterfly supported natively
- Large close-to-compute memories (L1)
- MIMD multiprocessors
- Fast, low-power, dynamic interconnect
- High IO bandwidth
- Large constant memory (nice-to-have)

ZPU architecture



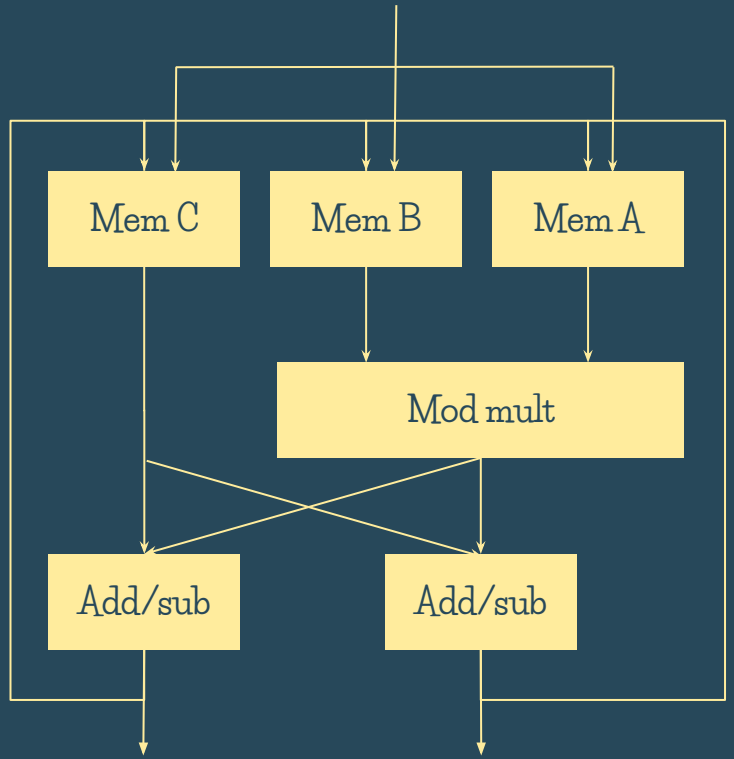
ZPU: the MIMD multiprocessor



- Dedicated MMU per MP
- Multi instruction multiple data program
- Full-bandwidth low-power remapping interconnect between PEs
- Large-integer wide buses between PEs and interconnect

ZPU: the processing element

- Close-to-compute memories
- Cooley-Tukey butterfly ALU
  - Large-integer modular multiplier
  - Multiply and dual accumulate
- Supports 64 to 384 bits natively



BLS12-377 base field mult in GPU

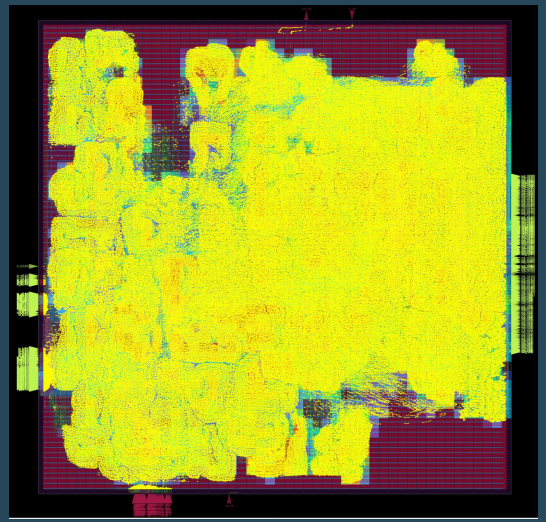
- NVIDIA 3090Ti
  - 628 mm^2 (Samsung 8 nm process)
  - 450W (w/ active cooling)
  - 10 TOPs (uint32 * uint32 → uint32)
- BLS12-377 base-field mult
  - Montgomery
  - # uint32 digits: n = 12
  - # req. ops: $5n^2 = 720$

- **Power: 1 Gmult/sec @ 30W**
- Consistent with benchmark results
  - ICICLE
  - others

BLS12-377 base field mult in ZPU

- OpenROAD (theopenroadproject.org)
  - ASAP7 PDK (7 nm academic)
  - Density factor 60%
  - Frequency 500 MHz
  - Untuned & unoptimized
- BLS12-377 base-field mult
  - **Barrett** & Karatsuba
  - Our Ultrascale+ design

- **Power: 1 Gmult/sec @ 2W**
- **Area: 0.233 mm^2** (mult only)

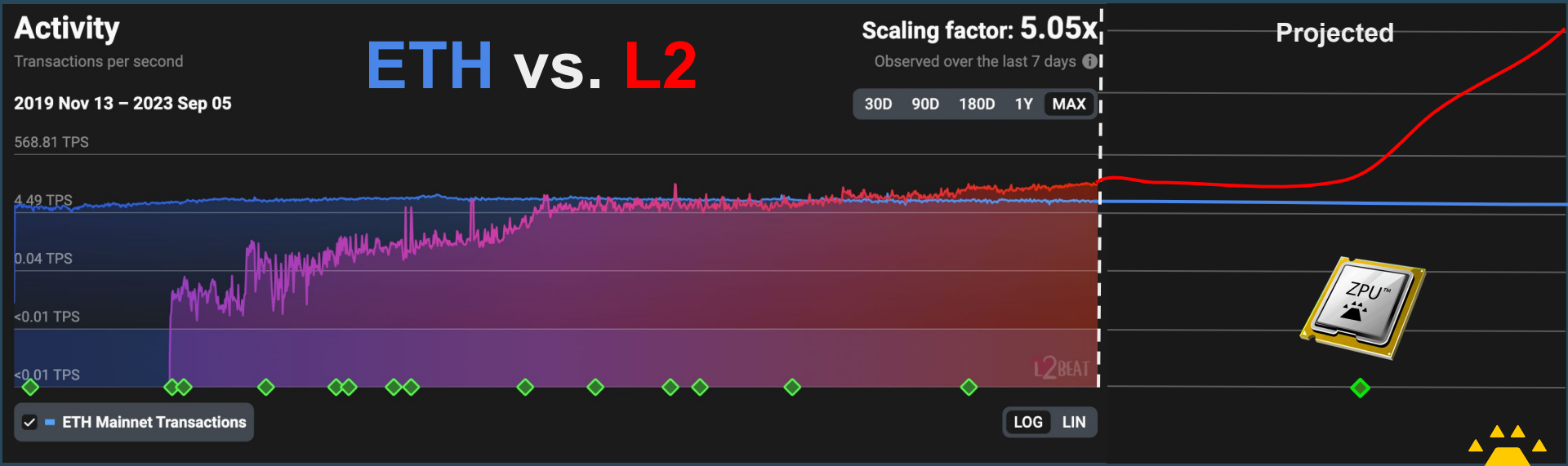


BLS12-377 base field mult in ZPU

- Currently 15 times more power efficient than GPU
  - Expecting approximately two orders of magnitude with tuning and optimization
- Significantly smaller die size
  - Increased yield, lower price
  - Passive cooling, better packing
- Dedicated arithmetics allows fewer PEs
  - Reduced control logic and dataflow, MIMD

Thank you

yuval@ingonyama.com



Sources: https://l2beat.com/scaling/activity