Performance with Halo2

•••

Aurélien Nicolas @ Scroll

2022-11-15

Circuit Engineering on Halo2

Thoughts and pointers for circuit designers.

Case study of the zkEVM by the Ethereum Foundation & Scroll.

Performance-related knobs

Witness Area

The amount of data and intermediate results.

Height / Width

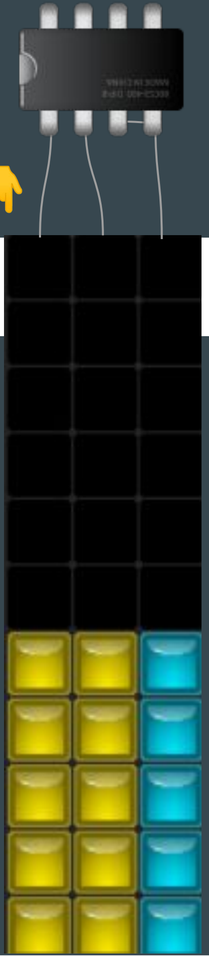
Prover versus verifier?

Recursion?

Gate Complexity

Implementation efficiency.

All gates repeat on every row. 👇



Witness Area - Data Structures First

Inputs, outputs, and intermediate results consume cells, arranged in a table.

Static patterns for simplicity. Dynamic for optimizations.

Alignment! 👉

EVM example: ADD two inputs into one output.

256 bits decomposition:	3 * 256 cells.
32 bytes decomposition:	3 * 32 cells.
Truncated decomposition:	fewer cells, data-dependent.

Gates (colors) are mutually exclusive when they share columns. 👇

Witness Area - Gates Wiring

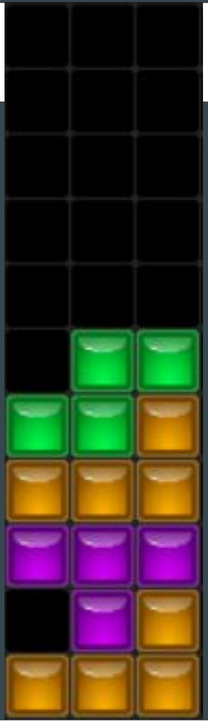
Independent gates may use **dedicated or shared columns**.

Mutually exclusive gates waste prover time; but minimize area.

The allocation can be data-dependent. Example: EVM steps have a variable shape; smaller ops consume fewer cells.

Measure the **utilization %** of gates and cells.

Cells are unused when nothing fits. 👉

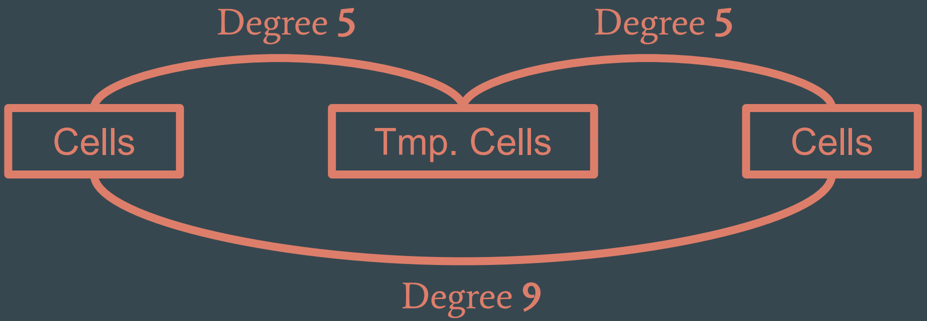


Gate Max Degree

Correlated with prover time. Start with **5**; otherwise 9. Unit-test it.

Most gates should exploit the max. Otherwise, **trade** lower degree for more witness rows. EVM: many simple operations on a large data path == low degree.

High degree expressions can be **split** via intermediate cells. This is automated in the zkEVM codebase (*split_expression*).



Height / Width

The main guideline is:

Height ~ prover cost ~ #rows * gate degree

Width ~ verifier cost ~ #columns

Variants of Halo2 have very different verifier costs. Pick:

- A polynomial commitment scheme (IPA, KZG, FRI).
- A multiopen protocol (Plonk, Shplonk, Halo).

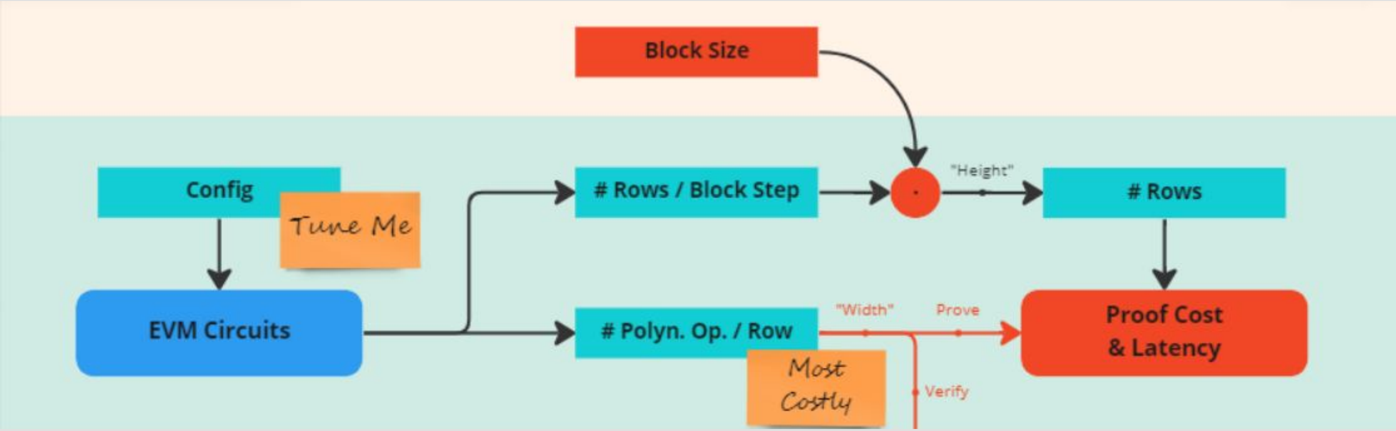
Original variant: IPA, no trusted setup, extra cost ~ #rows

Cost Model
zk-rollup

Large witness and many gates.

Minimize prover cost 📉

👉 Make as **wide** as possible.



Block Size

Config

Tune Me

EVM Circuits

Rows / Block Step

·

"Height"

Rows

Polyn. Op. / Row

"Width"

Prove

Most Costly

Verify

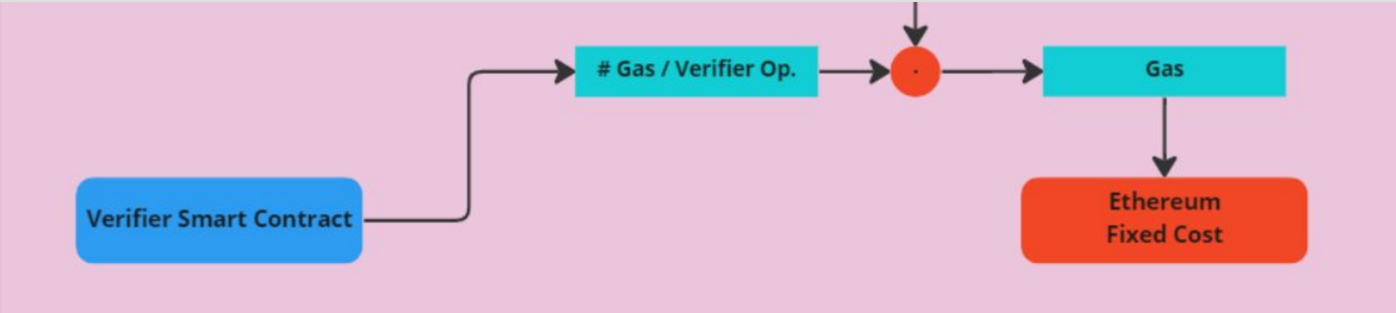
Proof Cost & Latency

Cost Model
zk-rollup

Costly on-chain verification.

👉 Must be **narrow**.

👉 Implies the KZG variant.



Verifier Smart Contract

Gas / Verifier Op.

·

Gas

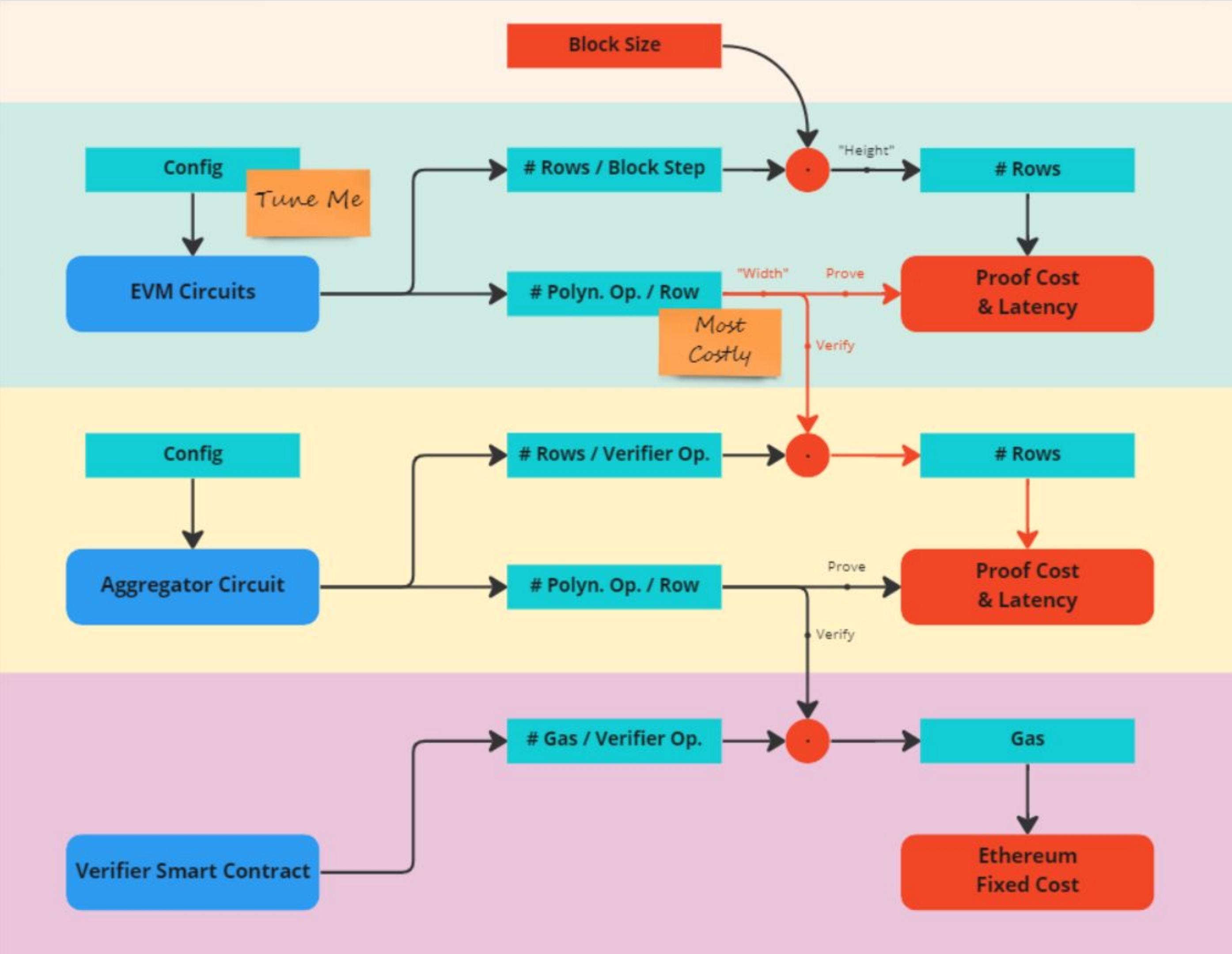
Ethereum Fixed Cost

Cost Model zk-rollup

Adapt wide to narrow.

👉 The **aggregation** stage.

👉 Non-native ECC of BN254 over BN254.

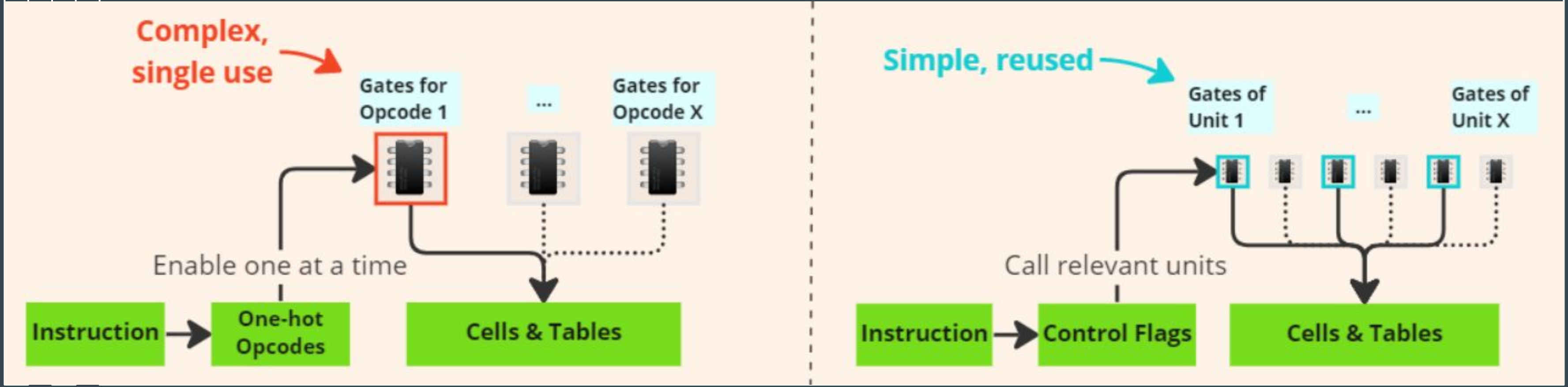


Gate Complexity - #multiplications

Factorization / DRY principle. Limited auto-optimization.

Small, composable gates, utilized in multiple cases. Route data paths to shared chips.

Clearer to think of circuits (vs. programming).



Gate Complexity

Exploit the power of lookups and (cheaper) multiset equalities:

ROM / function tables.

RAM / data bus / data routers.

They can replace a lot of arithmetic.

PLONK Standardization Workshop

ZK Engineering as accessible as regular software.

- Consolidating the IOP-based, modular approach to proof systems.
- The core: transcripts, commitments, zero-checks.
- The extensions and gadgets (lookups, ...).
- Witness and gates formats and APIs for DSLs.

Join us in the Breakout Room 1 at 2pm!