# ZK Light Client | April 2023 Roadmap

Design and implementation of test case # 4.3 (figure below) with recursive proof of epoch blocks and estimation of verification gas costs in EVM:

1. Designing a scheme for recursive proofs of epochal blocks in UML Activity diagram notation;
2. Software implementation of the scheme of recursive proofs of epochal blocks;
3. Testing the scheme of recursive proofs of epochal blocks on synthetic input data;
4. Preparation of a chain of epochal blocks from the NEAR blockchain;
5. Analysis of serialization algorithms for blocks and transactions from the NEAR blockchain;
6. Software implementation of recursive proof scheme with data serialization for blocks and transactions from the NEAR blockchain;
7. Testing the recursive proof scheme on the chain of epochal blocks from the NEAR blockchain;
8. Estimating the complexity and cost of publishing proofs of epochal blocks in the form of smart contracts in the public blockchain (EVM);
9. Optimizing the recursive epoch block proof scheme to minimize the complexity and cost of publishing a smart contract (take a slow/small plonky2 proof and invoke rapidsnark with it and its PIs, and we can follow the approach what PolymerDAO “plonky2-circom” does in its example for that);
10. Testing the optimized recursive epoch block proof scheme;
11. Preparation of report documentation describing the test case and software implementation.
12. Research on reducing circuits for computational integrity (sha256 hashing) of the chain. There is a secured possibility of leaving only the last hash computation in the circuit, instead of all 3 that a BlockHeader needs.

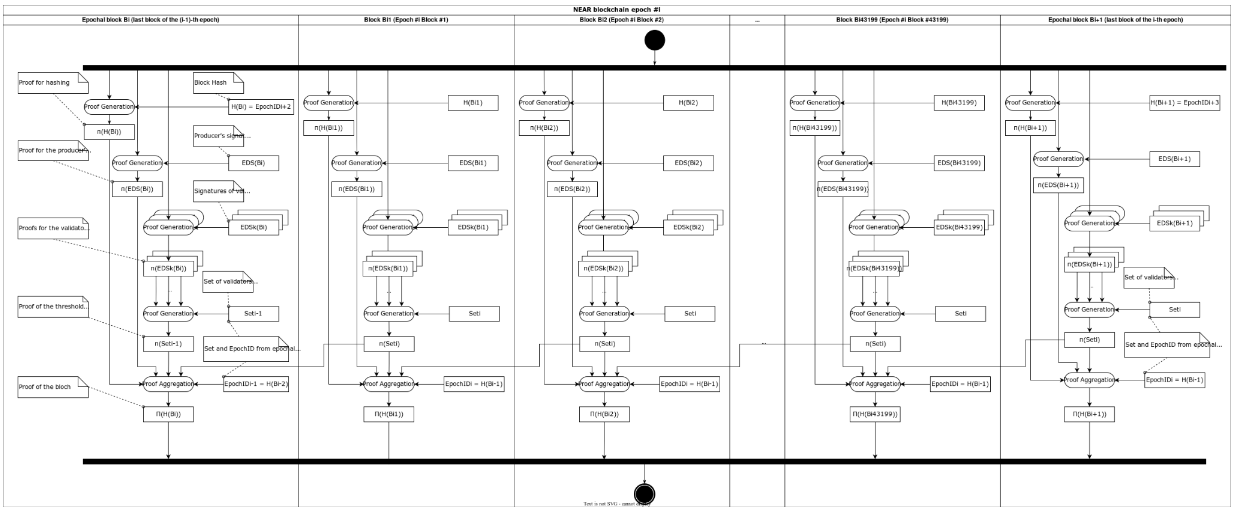

*Figure – Cryptographic chain of blocks*

The general scheme for building a chain of recursive proofs includes:

1. Formation of proofs $\pi(EDS_k(B_{ij}))$ of the CI of digital signatures $EDS_k(B_{ij})$ of validators of ordinary blocks. The list of block validators must match the list in $Set_{i-1}$ *of validators & producers* of the penultimate epoch;
2. Proof aggregation. Generation of proof of computational integrity of each block by aggregating:
   a. a proof of the correct hashing $\pi(H(B_{ij}))$ of the ordinary block $B_{ij}$;
   b. a proof of the correct digital signature $\pi(EDS(B_{ij}))$ of the block producer $B_{ij}$;
   c. $EpochId_{i-1}$ is an epoch identifier (constant);
   d. a proof of correct digital signatures $\pi(EDS_k(B_{ij}))$ of block validators $B_{ij}$.
   e. a final proof $\pi(EDS_k(B_{ij-1}))$ of the previous block $B_{ij-1}$.

Epoch block proofs additionally contain a proof of the correct hashing $\pi(H(B_{i-2}))$ of the epoch block $B_{i-2}$, i.e. the correctness of the calculation of the epoch identifier $EpochId_{i+2} = H(B_{i-2})$.

**Gantt chart**

| Weeks | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | | | | |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |

| | | | | |
|---|---|---|---|---|
| 6 | | | | |
| 7 | | | | |
| 8 | | | | |
| 9 | | | | |
| 10 | | | | |
| 11 | | | | |
| 12 | | | | |

| | |
|---|---|
| | Alexandr |
| | Kateryna |
| | Andrii |