**Project Description**

FPTreeDB is a RocksDB utility, optimized for Crystal Ridge, whose design is inspired by the paper "FPTree: A Hybrid SCM-DRAM Persistent and Concurrent B-Tree for Storage Class Memory." FPTreeDB bypasses the RocksDB LSM subsystem entirely, rather than attempting optimizations within the existing LSM implementation.

**Project Team:**

The prototype will be built and evaluated by the Crystal Ridge SW architecture team.

**Business Summary**

Key-value services are increasingly used for high-performance storage tiers in cloud architectures. An open-source reference implementation of a key/value datastore optimized for 3DXP could show performance and RAS advantages, creating pull for Crystal Ridge in the Public and Private cloud segments.

Depending on the results of this experiment, favorable business value might obtain in either of two ways: 1) FPTreeDB performance and 3DXP pricing could provide a superior economic alternative to LSM, if performance is high enough relative to cost or 2) FPTreeDB+3DXP could have moderately reduced performance/efficiency compared to LSM implementations, but with substantially reduced solution complexity.

**Market/Requester**

Crystal Software Architecture and joint NSG Pathfinding team with a goal to create optimized, reference applications for the CSP and Private Cloud segments.

**How will you know when you are done?**

- Negative result: If RocksDB and NVML libraries are fundamentally incompatible

- Negative result: If FPTreeDB cannot reproduce performance results from the original paper (a risk given authors made no reference implementation available)

- Negative result: If FPTreeDB solution complexity exceeds that of a LSM implementation

- Positive result: If performance and pricing position FPTreeDB favorably with LSM implementations (using existing K/V pathfinding workloads for comparison)

**Technical Description**

As a RocksDB utility, FPTreeDB does not modify the core RocksDB distribution, but only adds code at expected extension points. FPTreeDB follows SpatialDB and TransactionDB utilities, which provide high-level wrappers using the RocksDB API.

Unlike some other K/V pathfinding efforts, FPTreeDB runs completely in user-space and doesn't propose any hardware acceleration.

**Details**

**In Scope**

A backlog of issues assigned to the first Alpha milestone are available in GitHub: https://github.com/RobDickinson/fptreedb

**Out of Scope**

(needs review)

**Dependencies**

None for PoC evaluation.

**Resources & Schedule**

Rob to estimate

**Prototype Availability**

Rob to estimate