risc0 hashmap

process

- Start by understanding risc0 via the examples (multiply)
- Work out the vector example with “sorted” and get it working
  - This is to enable understanding how prover hints are populated and made accessible to the risc VM
- Understand the hashmap requirement
- Make some assumptions to reduce the scope of the hashmap problem
- Come up with an optimal algorithm under the above assumptions
- Integrate a simple hashmap into the prover code

Assumptions based on understanding

- We want to reduce the number of instructions / cycles in the zk-vm code because each instruction needs to be proven
- Eg: even though hashing is computationally fast due to cpu special instructions etc, it’s more complex to prove because of the number of raw assembly ops / instructions and the number of cycles it takes
- If the goal is to reduce the number of instructions generated in the riscV program etc, it needs to have sufficient hints
- Main prg can execute the entire program first natively to populate the hints based on the execution pattern
- Zkvm execution happens right after the main execution and takes advantage of the provided hints
- Zkvm needs to be secure and should be able to “verify” the hints provided to prevent malicious execution.
- The verification process needs to be fast enough that using the prover hints is beneficial

HashMap algorithm

- https://github.com/musitdev/sovrisc0prj1
- https://github.com/musitdev/sovrisc0prj1/blob/main/sovcore/README.md

Further work

- Use a LinearMap conditionally based on number of inserts and gets
- Handle duplicates / overwrites in an efficient manner
- Count cycles in vm execution
  - https://github.com/risc0/risc0/blob/main/risc0/zkvm/src/prove/mod.rs#L377-L378 (not working quite how we expected)
  - riscv-tools, spike, qemu
- Improvements in existing algorithm (reduce number of read cycles)