**Scratch Ideas in Work –**

1. Block diagram for simulator execution and data flow
2. Transition diagrams-block level to microarchitecture to software modules and processes (in Rust).
3. Tree for OPCODE walk through – Could be part of Block Leve data flow – initial block, probably a fastest way to categorize the “next instruction”.

Features: {

What does it mean to be accurate?

What does it mean to support speed?

What does it mean to have options?

Option Family 1: Performance family options could mean varying cache and register file sizes and instantiating subcomponents (co-processors) and memory hierarchy.

}

1. How about SaaS Model, essentially allowing for execution over cloud for this Simulator Model?
2. How would you run time control this simulator, purely GUI based control or would there be CLI based control supposed? Would having a Python based control embedded with simulator module to control it’s parameter be better?
3. Co-processors are not Functional Units across all standard cores (RISC-V Shakti).
4. What defines the state of the system:
   1. What all the processor sub components be inclusive in defining state of the system?
   2. How will memory hierarchy be modeled in terms of delay statistics – delay(x), delay(x+1), …
5. What is the cycle of instruction in RISC-V? This in order to instructions are put thru to yield the end result.
6. Is it necessary in ISS – behavioral model - to have cache warming mode for ROI (Region of Interest)?
7. Where would the trace files be created?
8. Is design space exploration a requirement for this simulator or just a BDD?
9. How is the memory modeled?
   1. How will the instruction memory be stored?
   2. Data transfer sizes?
10. Processor State Modeling.
11. Statistics Gathering.