Task group name: Cryptography Task Group
Extension or extension group name: Scalar Crypto
Component names: N/A
Group contributor name: IIT Madras
Task group liaison email: datkins@veritify.com
Group contributor liaison email: vasan.vs@gmail.com
Creation date: 2021-March-10
Last Modified Date: 2021-March-18
Version: 0.1

Requirements

What has to be done and what the end result is:

- Implement the architectural tests and coverage model for the RISC-V Scalar Cryptography extension.
- The tests will be generated using CTG
- Coverage model will be integrated into ISAC
- The tests will be run using the RISCOF framework on SAIL/Spike
- The tests must be compliant with the test format spec, and otherwise be suitable to merge into the riscv-arch-tests repository.
    - They will *replace* the tests donated by Imperas.
- The tests must fulfil the coverage goals described in the scalar crypto test plan document available at the location: https://github.com/riscv/riscv-crypto/blob/master/tests/compliance/test-plan-scalar.adoc other than the ones which are outside the scope of this SoW.
- All instructions which are part of the scalar cryptography extension must have tests. This includes both scalar-crypto specific instructions, and those borrowed from the Bitmanip extension.
    - For instructions where the behavior is identical for both Bitmanip and scalar crypto, the tests should be re-usable between the extensions. E.g. the rotate instructions.
      For instructions where scalar crypto requires only a subset of the functionality of the bitmanip instruction (e.g. grevi, shfli, unshfli), only the variants required by the scalar crypto extension must be tested. These tests are not expected to be re-usable between Bitmanip and scalar crypto.
- The entropy source extension must be tested:
    - All *architectectural* interactions with the CSR interfaces to the entropy source must be tested. This includes the “pollentropy” and “getnoise” pseudo instructions, and ensuring that other CSR access instructions behave correctly when accessing the `mentropy` and `mnoise` CSRs.
        - This includes checking for invalid state transitions, or trying to read `mentropy` while the `mnoise.NOISE_TEST` bit is set.
    - The non-deterministic SEED field of the `mentropy` CSR should be treated like an IO device, and its value must not be used to construct the test signature.

- Constant time execution (data independent execution latency) for instructions as outlined in the scalar crypto specification is *out of scope* for this SOW.

Deliverables

List of components and the changes expected:

- The complete set of architectural tests, as described in the requirements section above.
- Documentation on how the tests were generated.
- Coverage figures for the tests.

Acceptance criteria

List with measurable results defined:

- The tests must have been merged into the riscv-arch-tests github repository to qualify as “accepted”.
    - This implies they meet all of the requirements with respect to quality and completeness.
- The tests must run on Spike and/OR Sail.
    - IIT are not responsible for any fixes to Spike and Sail which their tests discover, but may volunteer them.
    - If both Sail and Spike are available, Spike and Sail test signatures must be in agreement.
    - Waivers/exceptions to this may be granted for instructions not yet implemented in Sail.

Projected timeframe (best guess):

S.No.	Item	Duration (Week #)
1	Bring on board one engineer/intern	Week 1
2	Familiarity of the K Extension by the engineer	Week 2
3	Cover points generation based on ISAC methodology for: AES & SM4 SHA256 & SM3 SHA512 & AES RV64	 Week 3 Week 4 Week 5

	Remaining RV64 instructions Bitmanip instructions	Week 6 Week 7 - 9
4	Review of cover points by TG	Week 10
5	Updation of cover points based on review comments	Week 11
6	Generation of tests for the finalized cover points after review through CTG	Week 12 - 13
7	CSR Mentropy / GetNoise source tests to be hard coded	Week 14 - 15
8	Integration and generation of final test report	Week 16

Sign off dates (done for every version > 1.0):

- Task group liaison sign-off date:
- Group contributor sign-off date: