30/04/2019

XCrypto

A General Purpose Cryptographic ISE for RISC-V

Ben Marshall, Daniel Page, Thinh Pham

University of Bristol
Computer Science Department

Background

Problem Statement

Can we extend RISC-V to make embedded devices faster and more efficient at cryptographic workloads, while enhancing the (side-channel) security of the system?

- Embedded / edge devices are being asked to *do more*
- Security is often a low priority design metric
- Side channel attacks are increasingly important
- Cryptographic algorithms are performance **and** security critical

Cryptographic Implementation with RISC-V

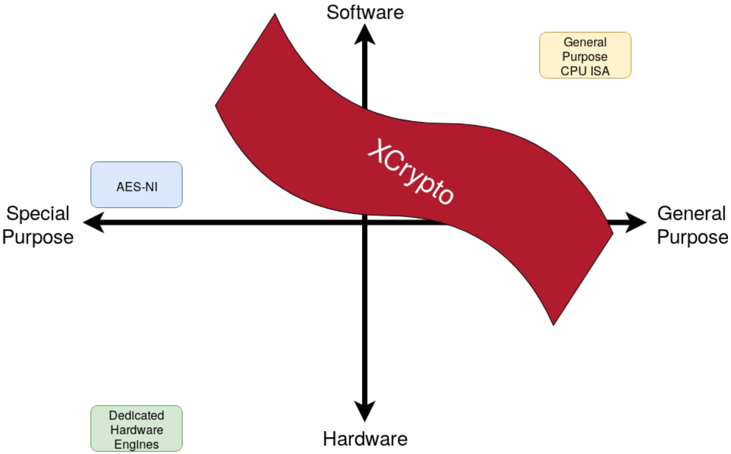
RISC-like architectures struggle with cryptographic workloads:

- Cryptographic operations are often niche, and hence slow without dedicated instructions. Examples include Galois Field and long integer arithmetic.
- They often require multiple operands to produce one result.
- Software based side-channel countermeasures are very expensive.

RISC-V can suffer next to ARM or MIPS:

- There is no bitwise rotate instruction (yet). This is *very* common in hash-functions and block ciphers.
- Overflow detection is (relatively) slow, which hinders long arithmetic operations for things like RSA.
- There are no indexed load/store instructions. More on this later.

Accelerating Cryptography



XCrypto:

XCrypto is *A solution* (v.s. *the solution*) to these problems, and an alternative to the standard RISC-V Crypto extension. As an ISE, it will not be the most efficient for any specific algorithm, but will be much better on average across a wide range of algorithms.

- **Light(er)weight:** It is designed to be suitable for micro-controllers, which the standard Crypto extension is not appropriate.
- **General purpose:** we show it improves a wide range of cryptographic workloads in terms of performance and code density.
- **Flexible:** It fits into multi-layered approaches to security, and has significant scope for side-channel countermeasure integration.
- **Experimental:** this is an on-going research project!

XCrypto: Extra State

XCrypto adds a 16 entry, 32-bit register file

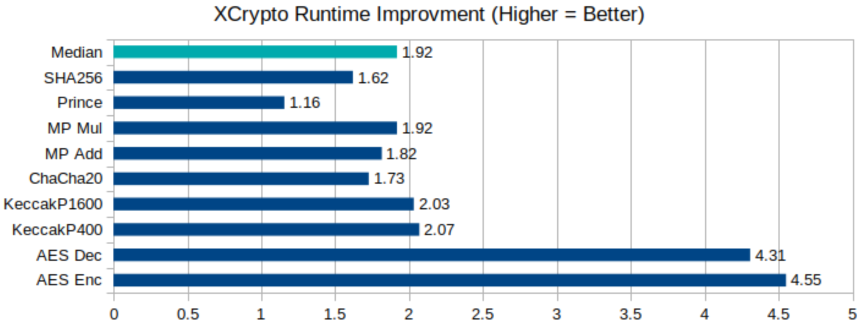
- Conceptually similar to a floating point co-processor, but for cryptography.
- Separate storage of cryptographic data
- Allows flexibility when integrating with existing cores
- Can combine XCrypto with RISC-V data-path or as a separate secure data-path.
- Expensive side-channel countermeasures can be applied only to XCrypto state.

XCrypto: New Instructions

XCrypto instructions are grouped into optional components:

- **Baseline ISE:** load/store to XCrypto registers, moves between GPRs/XCRs.
- **Randomness:** Seeding, sampling and quality checking of entropy source.
- **Memory:** Sub-word scatter/gather operations.
- **Bit:** permutations, bitfield insert/extract, lookup-tables
- **Packed:** SIMD-within-a-register (SWAR) arithmetic operations on 32/16/8/4/2 bit packed values.
- **Multi-precision:** support for long-integer arithmetic
- **AES:** lightweight AES sub-bytes and mix-columns acceleration
- **SHA3:** code-dense, energy efficient state index generation

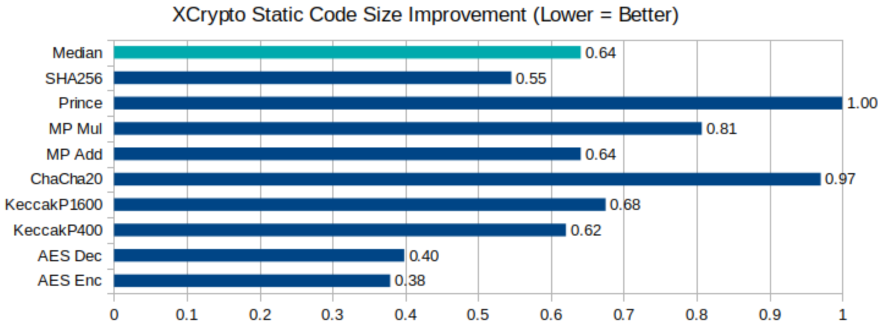
XCrypto: Runtime Improvement



As measured using the SPIKE ISA simulator and our area optimised reference implementation.

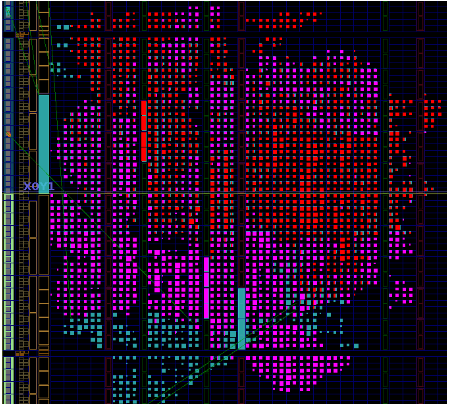
We use hand-optimised RISC-V assembly as a baseline.

XCrypto: Static Code Size



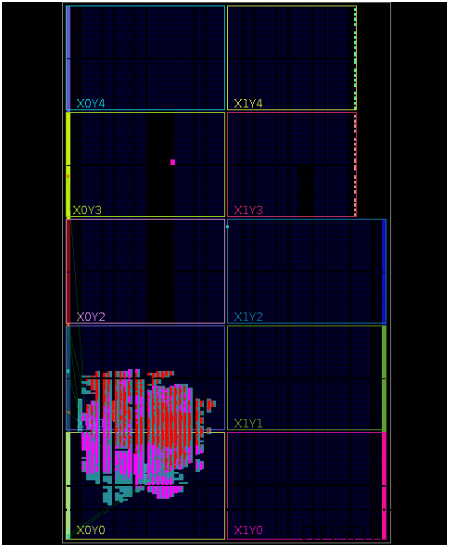
XCrypto: Reference Implementation

- Formally verified, area optimised reference implementation, coupled to a PicoRV32 CPU.
- Full System: 8.5K LUTs, 1.1K FFs, 288 DMEM
- XCrypto only: 3.6K LUTs, 226 FFs, 192 DMEM
- XCrypto area overhead: 40%.
- 100MHz working frequency.
- Critical path from inside PicoRV32 through XCrypto multiplier.



XCrypto: Reference Implementation

- Formally verified, area optimised reference implementation, coupled to a PicoRV32 CPU.
- Full System: 8.5K LUTs, 1.1K FFs, 288 DMEM
- XCrypto only: 3.6K LUTs, 226 FFs, 192 DMEM
- XCrypto area overhead: 40%.
- 100MHz working frequency.
- Critical path from inside PicoRV32 through XCrypto multiplier.



Future Work & Acknowledgements

- Still some instructions under consideration (`memcpy, memset`).
- Starting on a 5-stage pipelined implementation.
- Working towards a pipelined side-channel resistant implementation.
- Keen for general / specific feedback: come and say hi, or email.
  - ben.marshall@bristol.ac.uk
- We will be presenting XCrypto at the June RISC-V Workshop in Zurich!

- The ISE specification, simulator, toolchain and reference implementation is available at: **github.com/scarv/xcrypto**

- This work has been supported in part by EPSRC via grant EP/R012288/1, under the RISE (`http://www.ukrise.org`) programme.