Efficient Cryptography on the RISC-V Architecture

Ko Stoffelen

iCIS | Digital Security
Radboud University

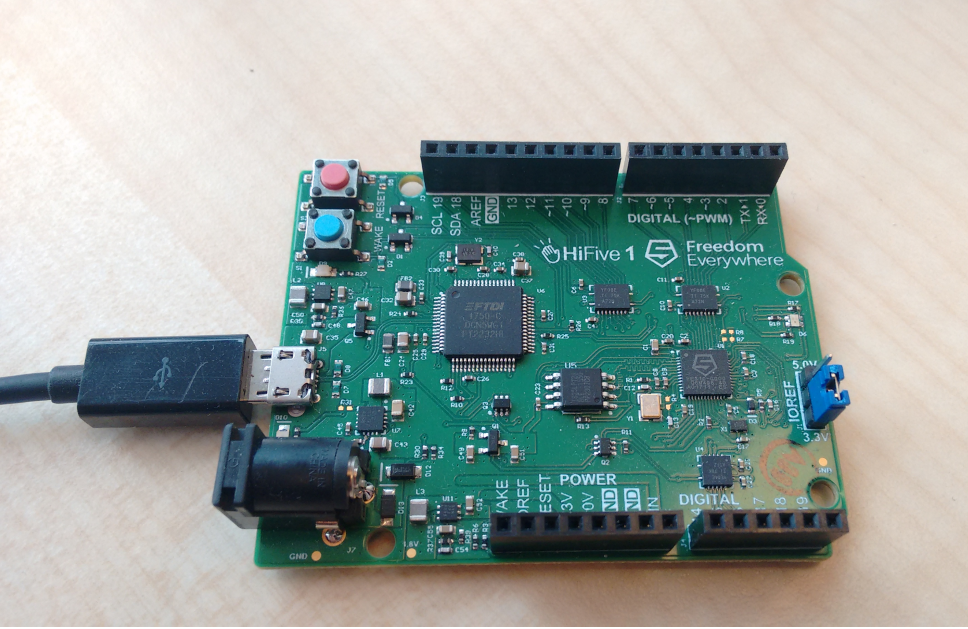
Tl;dr

In this talk:

- Fast AES-128 assembly for RV32I
- Fast ChaCha20 assembly for RV32I
- Fast Keccak-f[1600] assembly for RV32I
- Fast arbitrary-precision integer arithmetic for RV32IM
- Estimate potential speedup with several RISC-V extensions

RISC-V is . . .

- . . . a new open reduced instruction set architecture (ISA)
- . . . a research project that started at UC Berkely in 2010
- . . . a foundation with $>$ 325 members, including Google, Infineon, NXP, Qualcomm, Samsung, etc.
- . . . a serious competitor to ARM?
- . . . a big hype?
- . . . a project with a lot of work in progress
- . . . a frozen 32-bit base ISA (RV32I) and 64-bit (RV64I) with standardized optional extensions
- . . . a production-ready core design


HiFive1
Freedom Everywhere
DIGITAL (~PWM)
POWER
DIGITAL
IOREF
5.0V
3.3V
RESET
WAKE
SCL 19
SDA 18
AREF
GND

RV32I

- 32 32-bit registers `x0`–`x31`, but some are reserved
- Basic three-operand arithmetic and bitwise instructions
- Basic shift instructions
- Basic load/store instructions
- Basic jump, conditional jumps, comparison instructions
- That's more or less it — boring!
- No: rotate instructions, carry flag, DSP/vector instructions, nice bit operation instructions
- Compensated by extensions: `M`, `A`, `F`, `D`, `Q`, `C`, ...
  - `M`: integer multiplication/division
  - `B`: bit operation instructions (WIP)
- HiFive1: 5-stage single-issue in-order pipelined RV32IMAC E31 CPU
  - $<$ 384 MHz, 64 KiB RAM, 16 MiB flash, 16 KiB I$
  - Most instructions single cycle result latency, except loads

AES-128

- Lookup tables or bitsliced?
- Both! Depends on data caches
- 4 KiB table-based fairly straightforward [BS08]
  - Baseline of 704 instructions
  - `LBU` byte loads: ✓ (−4)
  - Everything else: ✗
  - Can't load from address with offset in registers (+160)
  - Key expansion in 340 cycles, encryption in 57 cycles/byte
- Bitsliced based on Cortex-M3/M4 implementation [SS16]
  - 2 blocks in parallel in CTR mode
  - RV32I advantage: no spills in SubBytes, enough registers!
  - RV32I disadvantage: no rotates, no byte extraction
  - Key expansion in 1239 cycles, encryption in 124.4 cycles/byte

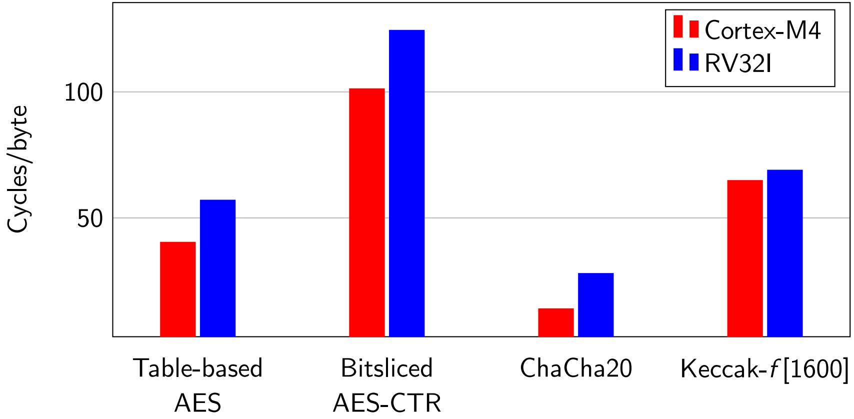
ChaCha20

- Stream cipher with 512-bit state
- RV32I advantage: fits in registers
- RV32I disadvantage: no rotates
- Encryption/decryption in 27.9 cycles/byte

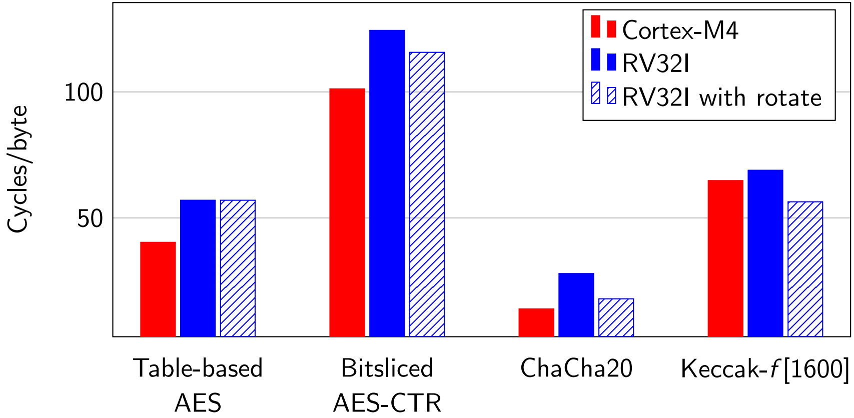
Keccak-f[1600]

- Design space explored in Keccak Implementation Overview [BDP+12]
  - Bit interleaving: ✓
  - Lane complementing: ✓
  - State extension for smoother scheduling: ✓
  - Plane per plane: ✓
  - In-place: ✓
- Inspired by Cortex-M3/M4 implementation in XKCP
- Permutation in 72.4 cycles/byte

Speed comparison



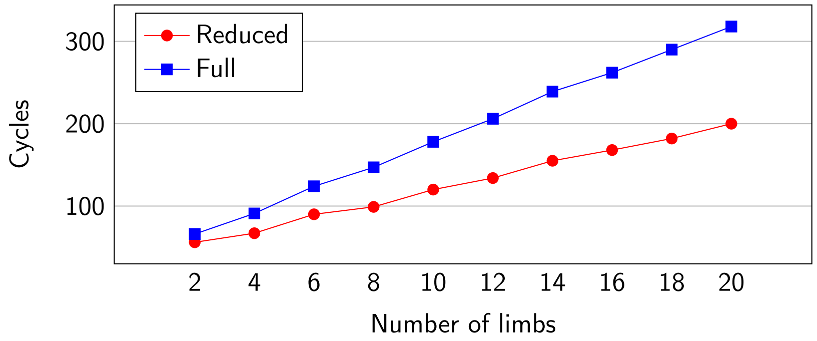
What if. . . single-cycle rotations?



Arbitrary-precision arithmetic

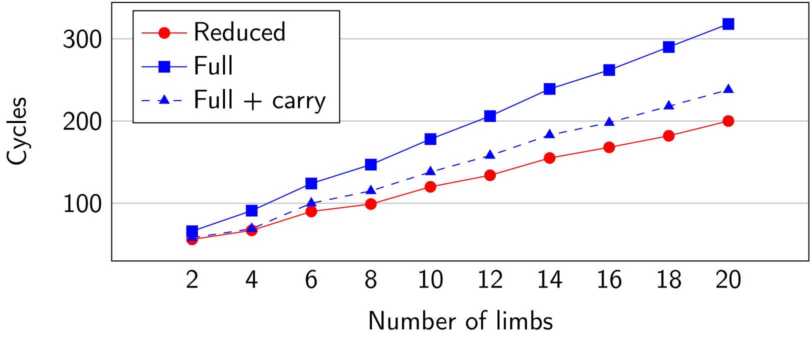
- A.k.a. big-integer arithmetic (well, only $+$, $\times$)
- Used by RSA, ECC, some post-quantum, ...
- Split large number in 32-bit *limbs*
- Addition of two 32-bit limbs may overflow
- RV32I: no carry flag!
- `ADDS r0,a0,b0`; `ADC r1,a1,b1` on ARM becomes
- `ADD r0,a0,b0`; `SLTU c,r0,a0`; `ADD r1,a1,b1`; `ADD r1,r1,c`
- Reduced-radix representations appear attractive
- Radix 2^k: only fill $k < 32$ bits per limb
- We keep it generic and don't fix specific radix

Arbitrary-precision addition



Note: reduced radix requires more limbs

What if... carry flag?

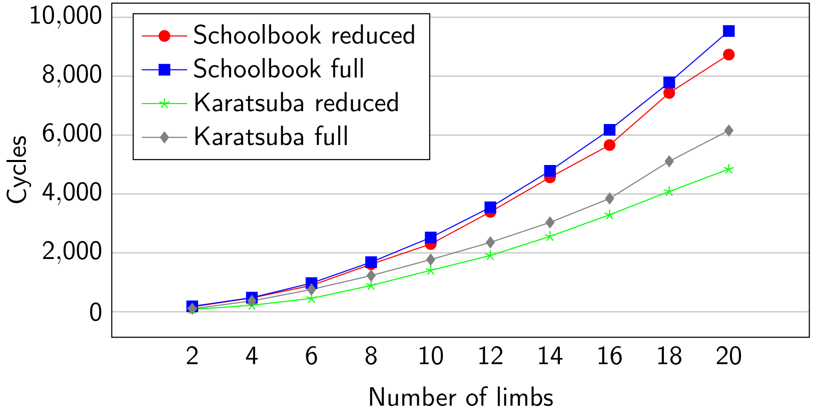


Note: reduced radix requires more limbs

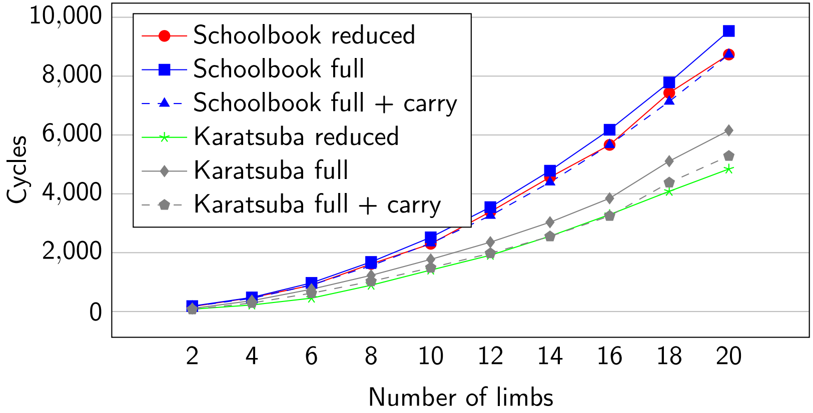
Arbitrary-precision multiplication

- `M` extension provides `MUL`/`MULHU` instructions
- Result latency of 2 cycles
- Consider schoolbook and one level of (subtractive) Karatsuba
- Instead of n-limb multiplication, do $3\left(\frac{n}{2}\right)$ multiplication and some additions/subtractions

Arbitrary-precision multiplication



What if. . . carry flag?



Some conclusions

- The base RV32I ISA is not that interesting for optimization
- Comparing speed results across different RISC-V cores is going to be a pain in the future
  - More variation in clock cycle behavior
  - Different standardized and perhaps also proprietary extensions
- Symmetric crypto would really benefit from nice bit operation instructions
- Carry-chain crypto would really benefit from a carry flag
- Having more registers is always nice

Thanks. . .

. . . for your attention!

Slides/paper at `https://ko.stoffelen.nl`

Code at `https://github.com/Ko-/riscvcrypto`

References I

Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche, and Ronny Van Keer.
Keccak implementation overview, May 2012.
https://keccak.team/files/Keccak-implementation-3.2.pdf.

Daniel J. Bernstein and Peter Schwabe.
New AES software speed records.
In Dipanwita Roy Chowdhury, Vincent Rijmen, and Abhijit Das, editors, *Progress in Cryptology - INDOCRYPT 2008: 9th International Conference in Cryptology in India*, volume 5365 of *Lecture Notes in Computer Science*, pages 322–336. Springer, Heidelberg, December 2008.

Peter Schwabe and Ko Stoffelen.
All the AES you need on Cortex-M3 and M4.
In Roberto Avanzi and Howard M. Heys, editors, *SAC 2016: 23rd Annual International Workshop on Selected Areas in Cryptography*, volume 10532 of *Lecture Notes in Computer Science*, pages 180–194. Springer, Heidelberg, August 2016.