RISC-V Instruction Set Extensions for Multi-Precision Integer Arithmetic

Hao Cheng[1], Georgios Fotiadis[1], Johann Großschädl[1], Daniel Page[2], Thinh Pham[2], Peter Y. A. Ryan[1]

[1] University of Luxembourg [2] University of Bristol

A case study on CSIDH-512

CSIDH: post-quantum isogeny-based key exchange

- Action of ideal class group on supersingular ECs
- Perform modular operations with MPIs at the lowest level
- CSIDH-512 NIST security level 1 and prime p 511 bits

Alice	Bob
$sk_A = (e_1, ..., e_n)$	$sk_B = (e'_1, ..., e'_n)$
$pk_A = \mathfrak{a} \star E_0 = E_A$	$pk_B = \mathfrak{b} \star E_0 = E_B$
$E_A \rightarrow$	
	$\leftarrow E_B$
$ss = \mathfrak{a} \star E_B = a \star (b \star E_0)$	$ss = \mathfrak{b} \star E_A = b \star (a \star E_0)$

Four different implementations of CSIDH-512

- Full-radix (64 bits/digit) vs. Reduced-radix (57 bits/limb)
- ISA-only (RV64GC) vs. ISE-supported (RV64GC + ISE)

ISA-only implementation

Focus on low-level optimization ($\mathbb{F}_p$ multiplication)

- Main building block is *Multiply-and-ACcumulate (MAC)*

$$S \leftarrow S + a_i \cdot b_j$$

- Full-radix 8 instr.

```
/* Input/Output: 192-bit accumulator  e || h || l */
/* Input:         64-bit operands     a and b     */
mulhu z, a, b; mul y, a, b; add  l, l, y; sltu y, l, y;
add   z, z, y; add h, h, z; sltu z, h, z; add  e, e, z;
```

- Reduced-radix 6 instr. + implicit overhead (more MACs)

```
/* Input/Output: 128-bit accumulator  h || l  */
/* Input:         64-bit operands     a and b */
mulhu z, a, b; mul y, a, b; add  l, l, y; sltu y, l, y;
add   z, z, y; add h, h, z;
```

ISE design

Full-radix

- `maddlu`: $rd \leftarrow (rs1 \times rs2 + rs3)\&(2^{64} - 1)$
- `maddhu`: $rd \leftarrow ((rs1 \times rs2 + rs3) \gg 64)\,\&(2^{64} - 1)$
- `cadd`: $rd \leftarrow ((rs1 + rs2) \gg 64) + rs3$

Reduced-radix

- `madd57lu`: $rd \leftarrow \big((rs1 \times rs2)\&(2^{57} - 1)\big) + rs3$
- `madd57hu`: $rd \leftarrow \big(((rs1 \times rs2) \gg 57)\,\&(2^{64} - 1)\big) + rs3$
- `sraiadd`: $rd \leftarrow rs1 + \mathrm{EXTS}(rs2 \gg imm)$

ISE-supported implementation

Impact of ISE

- Full-radix MAC now 4 instr.

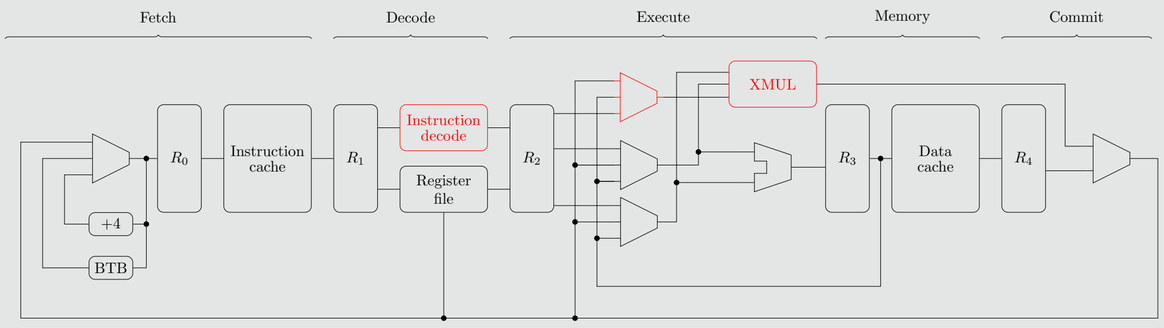
```
/* Input/Output: 192-bit accumulator  e || h || l */
/* Input:         64-bit operands     a and b     */
maddhu z, a, b, l; maddlu l, a, b, l;
cadd   e, h, z, e; add    h, h, z;
```

- Reduced-radix MAC now 2 instr. + accumulator aligned

```
/* Input/Output:  64-bit accumulators h and l */
/* Input:         64-bit operands     a and b */
madd57hu h, a, b, h; madd57lu l, a, b, l;
```

Hardware implementation of ISE

- RV64GC Rocket core (on Xilinx Artix-7 FPGA)
- XMUL: 3rd input operand + custom instructions



Evaluation

Hardware: both full/reduced-radix $\sim$ 10% overhead

	LUTs	Regs	DSPs	CMOS
Base core	4807	2156	16	428680
Base core + ISE (full-radix)	5019	2390	16	483248
Base core + ISE (reduced-radix)	5223	2352	16	495290

Software: cycle count

- Full-radix *faster* in ISA-only
- Reduced-radix *more suitable* for ISE-supported

Operation	Full-radix		Reduced-radix	
	ISA-only	ISE-sup.	ISA-only	ISE-sup.
integer multiplication	608	371	625	303
integer squaring	440	371	398	216
Montgomery reduction	730	469	818	389
fast modulo-p reduction	107	107	112	104
$\mathbb{F}_p$ addition	163	163	148	132
$\mathbb{F}_p$ subtraction	143	143	139	123
$\mathbb{F}_p$ multiplication	1446	954	1561	799
$\mathbb{F}_p$ squaring	1279	951	1334	712
CSIDH group action	701.0 M	502.9 M	736.2 M	411.1 M
	1.00×	1.39×	0.95×	1.71×

This work was supported, in part, by the Lux4QCI project, the Luxembourg FNR via PABLO project (grant ref. 16326754), the EPSRC through grant EP/R012288/1, and Innovate UK through project 10065634 (SCHEME).