PERCIVAL: Integrating Posit and Quire Arithmetic into the RISC-V Ecosystem

David Mallasén, Raul Murillo, Alberto A. Del Barrio, Guillermo Botella, Luis Piñuel, Manuel Prieto-Matias

dmallasen@ucm.es

Department of Computer Architecture and Automation
Computer Science and Engineering Faculty
Complutense University of Madrid

UNIVERSIDAD COMPLUTENSE MADRID

RISC-V Summit Europe 2023
June 6-8, 2023

Overview

Real-Number Arithmetic

How do we represent 1.5, $\sqrt{2}$ or π?

Real-Number Arithmetic

How do we represent 1.5, $\sqrt{2}$ or π?

- IEEE 754 floating-point representation

Real-Number Arithmetic

How do we represent 1.5, $\sqrt{2}$ or π?

- IEEE 754 floating-point representation

Emerging alternatives:

- Google's bfloat16
- Nvidia's TensorFloat

Real-Number Arithmetic

How do we represent 1.5, $\sqrt{2}$ or π?

- IEEE 754 floating-point representation

Emerging alternatives:

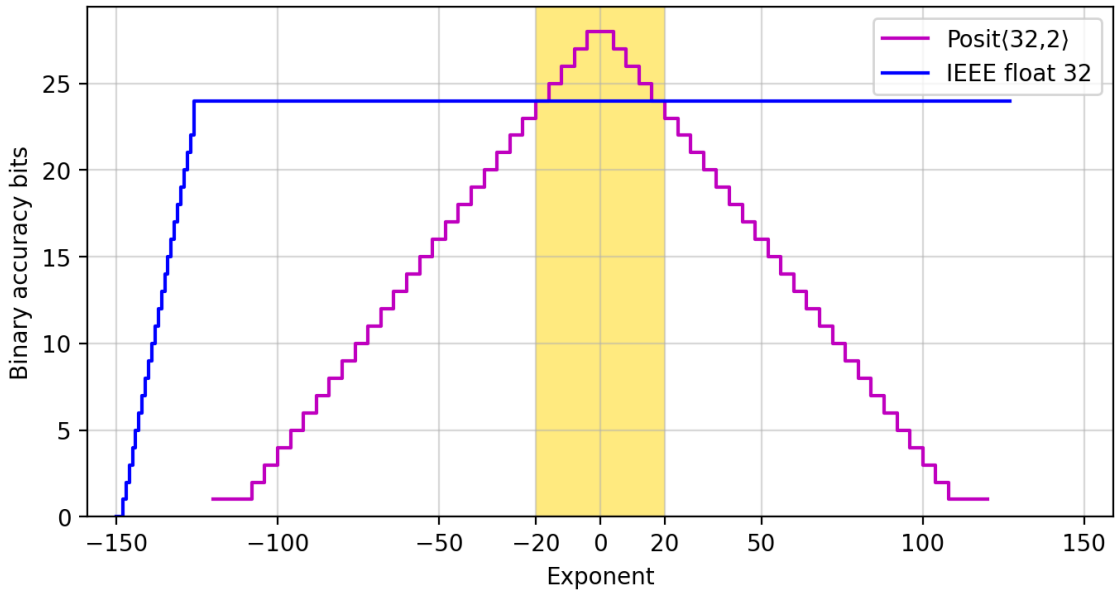
- Google's bfloat16
- Nvidia's TensorFloat
- Posit arithmetic

Posit Arithmetic

- Variable-length regime
- Variable-length fraction
- Trade-off:
  accuracy - dynamic range

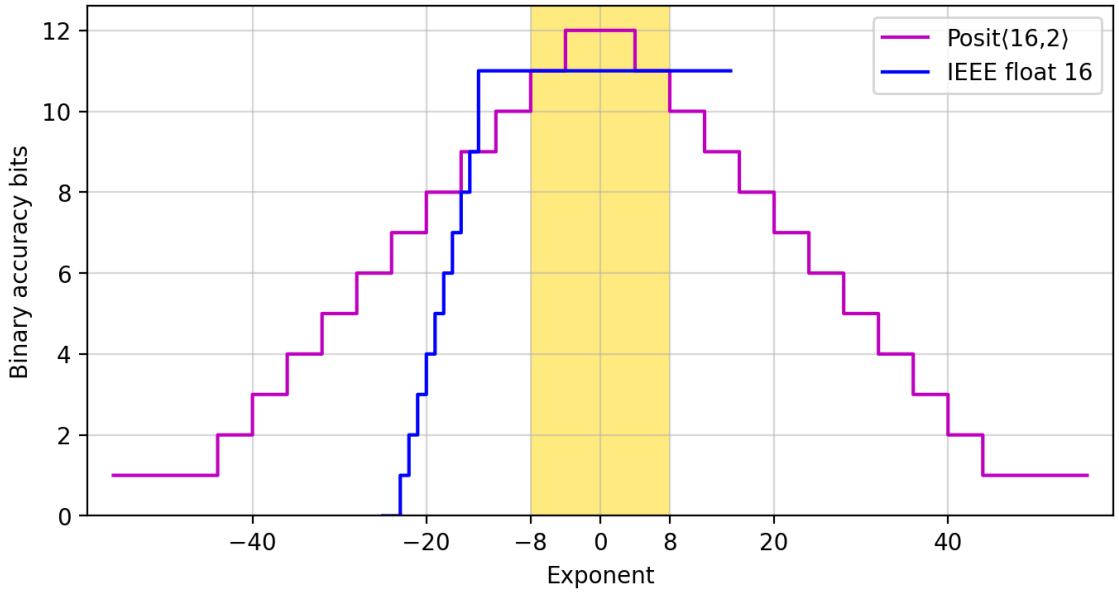
Posit Arithmetic

- Variable-length regime
- Variable-length fraction
- Trade-off:
  accuracy - dynamic range



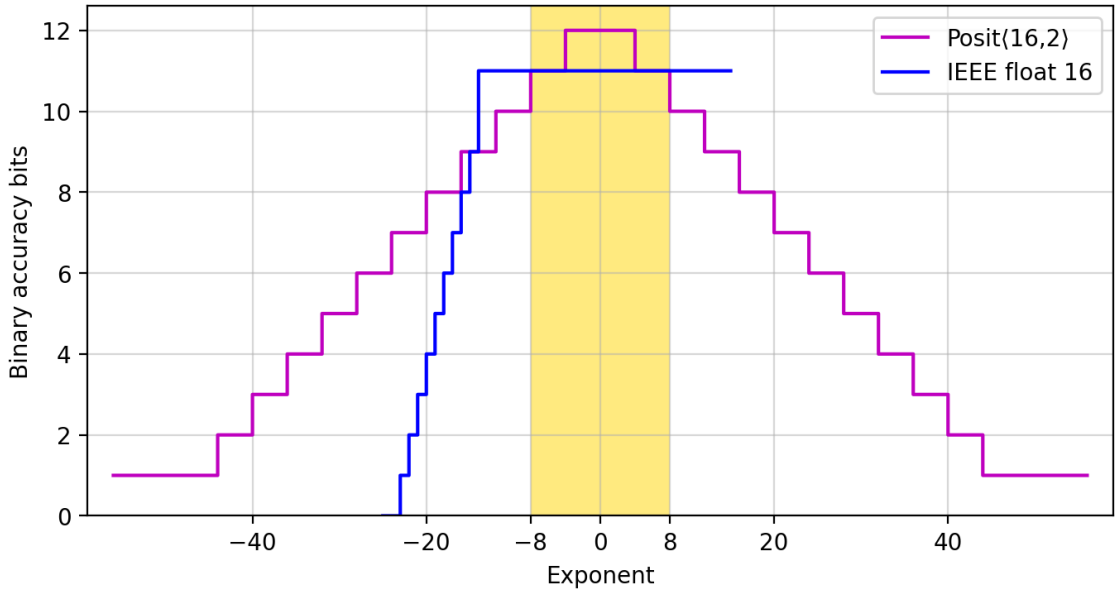
Posit Arithmetic

- Variable-length regime
- Variable-length fraction
- Trade-off:
  accuracy - dynamic range



Posit Arithmetic

- Variable-length regime
- Variable-length fraction
- Trade-off:
  accuracy - dynamic range
- **Quire accumulator** register
- $2^{31} - 1$ fused MAC without **accuracy** loss



PERCIVAL Posit CPU

Based on CVA6 (PULP's Ariane) maintained by the OpenHW Group.

- RV64I: Base integer operations
- M: Integer multiply/divide
- A: Atomic memory operations
- **F:** Single-precision IEEE 754 FP
- **D:** Double-precision IEEE 754 FP
- C: Compressed instructions

PERCIVAL Posit CPU

Based on CVA6 (PULP's Ariane) maintained by the OpenHW Group.

- RV64I: Base integer operations
- M: Integer multiply/divide
- A: Atomic memory operations
- **F:** Single-precision IEEE 754 FP
- **D:** Double-precision IEEE 754 FP
- C: Compressed instructions

- **Xposit:** Posit32 + quire instructions

Xposit RISC-V Custom Extension

Instructions for 32-bit posits and quire:

- Load/Store
- Computational
- Quire
- Conversion posit-integer
- Register move
- Comparison

Xposit RISC-V Custom Extension

Instructions for 32-bit posits and quire:

- Load/Store
- Computational
- Quire
- Conversion posit-integer
- Register move
- Comparison

Compiler support

We modified the LLVM compiler adding Xposit.
Allows to compile C programs together with posit instructions.

PERCIVAL Posit CPU

Based on CVA6 (PULP's Ariane) maintained by the OpenHW Group.

- RV64I: Base integer operations
- M: Integer multiply/divide
- A: Atomic memory operations
- **F:** Single-precision IEEE 754 FP
- **D:** Double-precision IEEE 754 FP
- C: Compressed instructions

- **Xposit:** Posit32 + quire instructions

Prime platform for comparisons between posits + quire and IEEE 754 FP

PERCIVAL Posit CPU

Based on CVA6 (PULP's Ariane) maintained by the OpenHW Group.

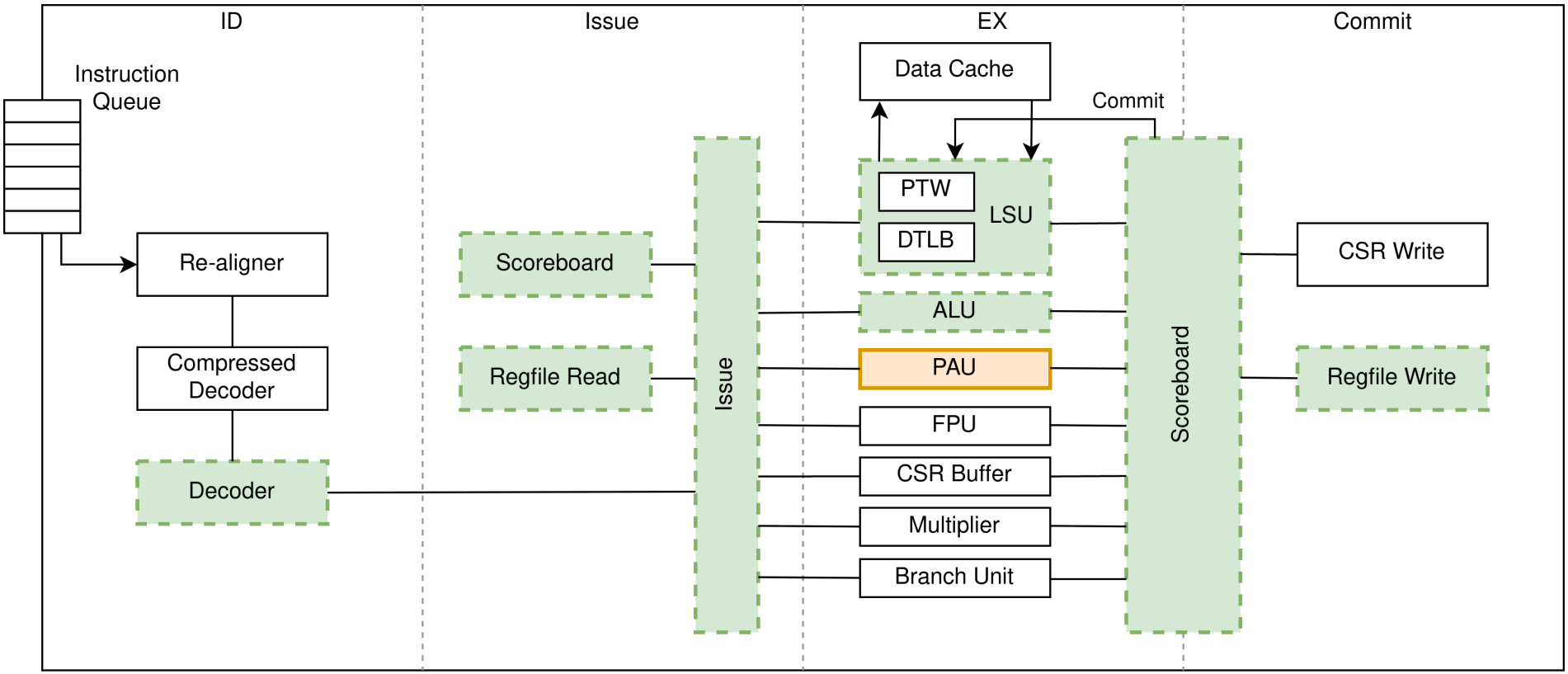
- RV64I: Base integer operations
- M: Integer multiply/divide
- A: Atomic memory operations
- **F:** Single-precision IEEE 754 FP
- **D:** Double-precision IEEE 754 FP
- C: Compressed instructions
- **Xposit:** Posit32 + quire instructions

Prime platform for comparisons between posits + quire and IEEE 754 FP

Posit operations **side by side** with floating-point.

- Posit Arithmetic Unit (PAU)
- Posit register file
- Native load and store support

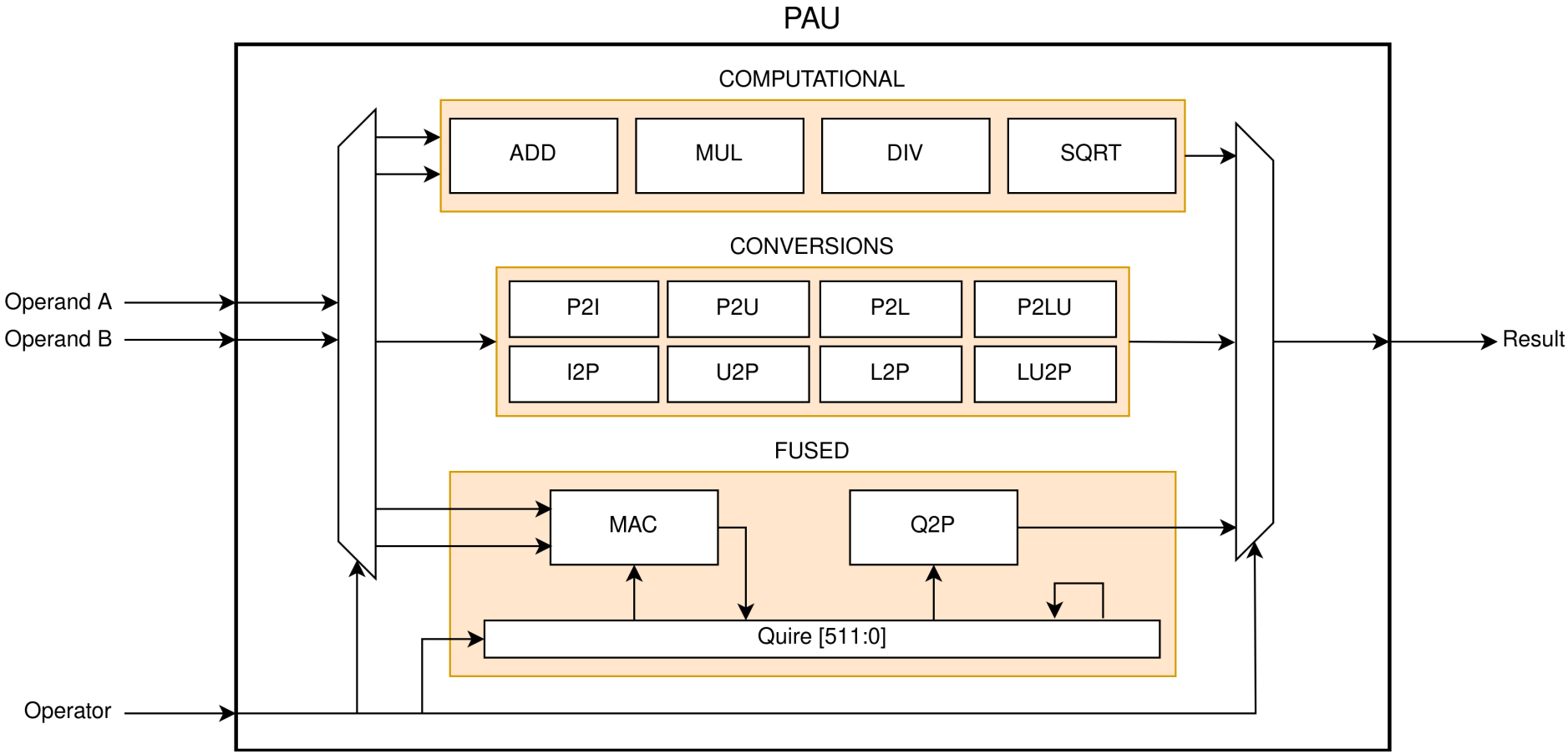
PERCIVAL Diagram



D. Mallasén, R. Murillo, A. A. D. Barrio, G. Botella, L. Piñuel, and M. Prieto-Matias, "PERCIVAL: Open-Source Posit RISC-V Core With Quire Capability," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 3, pp. 1241–1252, 2022, doi: 10.1109/TETC.2022.3187199

Posit Arithmetic Unit (PAU)

- Multi-cycle operations
- Similar latency to FPU (FPNew)
- Synchronous handshake interface



FPGA Synthesis Results

- PERCIVAL runs on a Xilinx Kintex-7 FPGA (Genesys II)
- Target frequency set by CVA6: 50 MHz

FPGA Synthesis Results

- PERCIVAL runs on a Xilinx Kintex-7 FPGA (Genesys II)
- Target frequency set by CVA6: 50 MHz

32-bit	LUTs	FFs	DSP blocks
FPU	4046	973	2
PAU	11879	2985	4

- 32-bit posit with 512-bit quire:
  - ~3x more resources than FP

FPGA Synthesis Results

- PERCIVAL runs on a Xilinx Kintex-7 FPGA (Genesys II)
- Target frequency set by CVA6: 50 MHz

32-bit	LUTs	FFs	DSP blocks
FPU	4046	973	2
PAU	11879	2985	4

- 32-bit posit with 512-bit quire:
  - ~3x more resources than FP

Name	LUTs	FFs
PAU top	593	1063
Posit Add	784	106
Posit Mult	736	73
Posit ADiv	413	43
Posit ASqrt	426	33
Quire MAC	5644	1541
Quire to Posit	889	126
Int to Posit	176	0
Long to Posit	331	0
ULong to Posit	425	0
Posit to Int	499	0
Posit to Long	379	0
Posit to UInt	228	0
Posit to ULong	358	0
PAU total	11879	2985
PAU w/o quire	5346	1318

FPGA Synthesis Results

- PERCIVAL runs on a Xilinx Kintex-7 FPGA (Genesys II)
- Target frequency set by CVA6: 50 MHz

32-bit	LUTs	FFs	DSP blocks
FPU	4046	973	2
PAU	11879	2985	4

- 32-bit posit with 512-bit quire:
  - ~3x more resources than FP

Name	LUTs	FFs
PAU top	593	1063
Posit Add	784	106
Posit Mult	736	73
Posit ADiv	413	43
Posit ASqrt	426	33
Quire MAC	**5644**	**1541**
Quire to Posit	**889**	**126**
Int to Posit	176	0
Long to Posit	331	0
ULong to Posit	425	0
Posit to Int	499	0
Posit to Long	379	0
Posit to UInt	228	0
Posit to ULong	358	0
PAU total	11879	2985
PAU w/o quire	**5346**	**1318**

FPGA Synthesis Results

- PERCIVAL runs on a Xilinx Kintex-7 FPGA (Genesys II)
- Target frequency set by CVA6: 50 MHz

32-bit	LUTs	FFs	DSP blocks
FPU	4046	973	2
PAU	11879	2985	4

- 32-bit posit with 512-bit quire:
  - ~3x more resources than FP

- Quire MAC uses ~50% of the area
- 32-bit posit without quire:
  - ~33% more resources than FP

Name	LUTs	FFs
PAU top	593	1063
Posit Add	784	106
Posit Mult	736	73
Posit ADiv	413	43
Posit ASqrt	426	33
Quire MAC	**5644**	**1541**
Quire to Posit	**889**	**126**
Int to Posit	176	0
Long to Posit	331	0
ULong to Posit	425	0
Posit to Int	499	0
Posit to Long	379	0
Posit to UInt	228	0
Posit to ULong	358	0
PAU total	11879	2985
PAU w/o quire	**5346**	**1318**

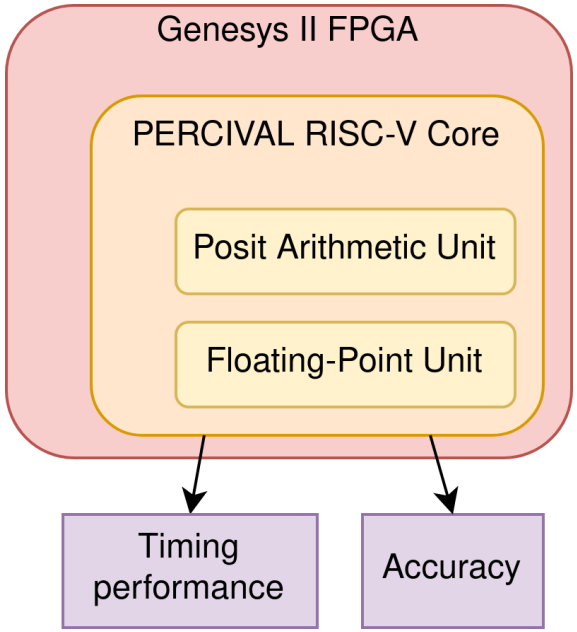
Benchmarks

Software benchmarks:

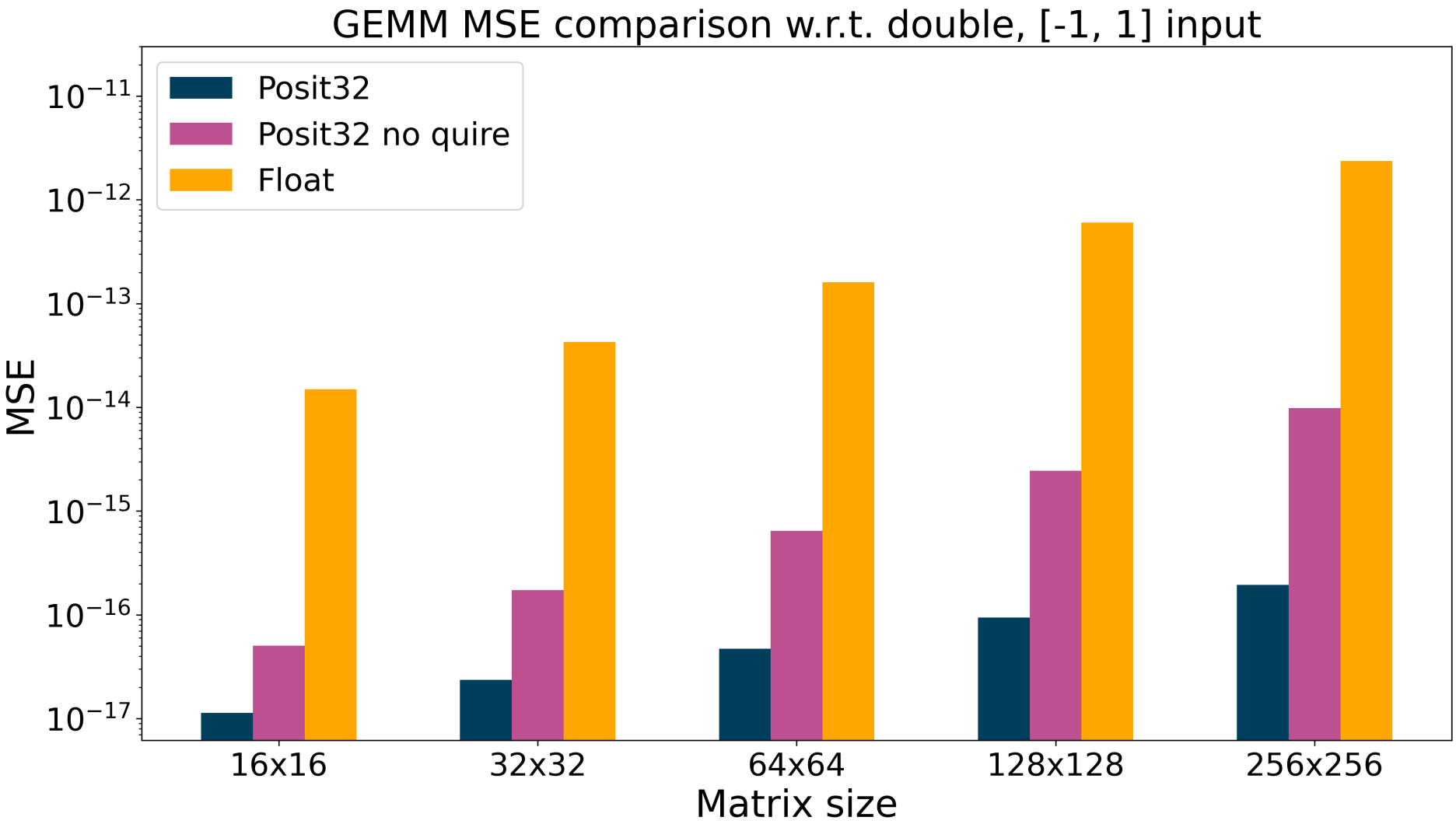
- General Matrix Multiplication (GEMM) $\rightarrow$ Posit32
- Cholesky $\rightarrow$ Posit64

Metrics:

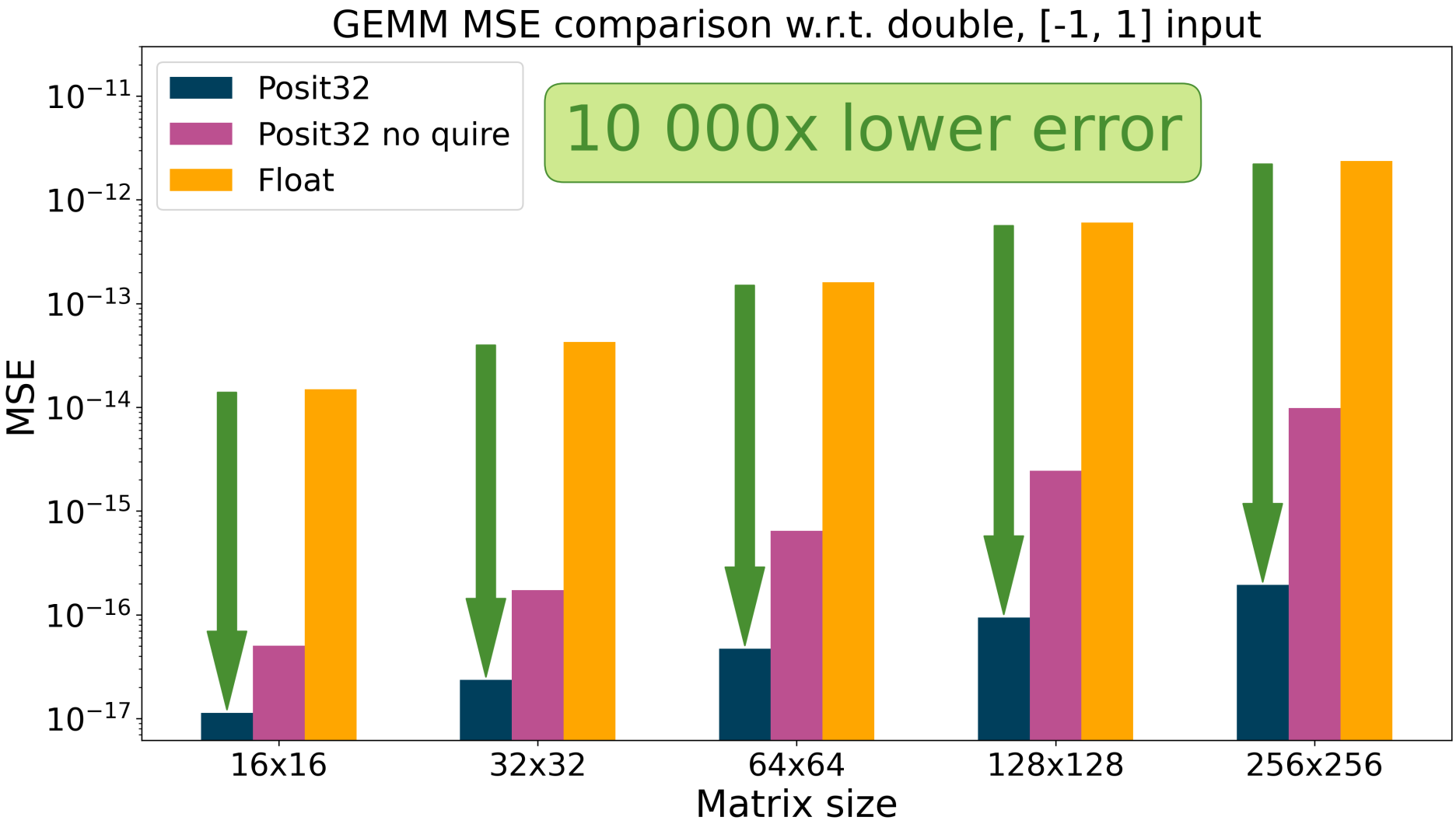
- Performance: FPGA runtime
- Accuracy:
  - Mean Squared Error
  - Maximum Absolute Error



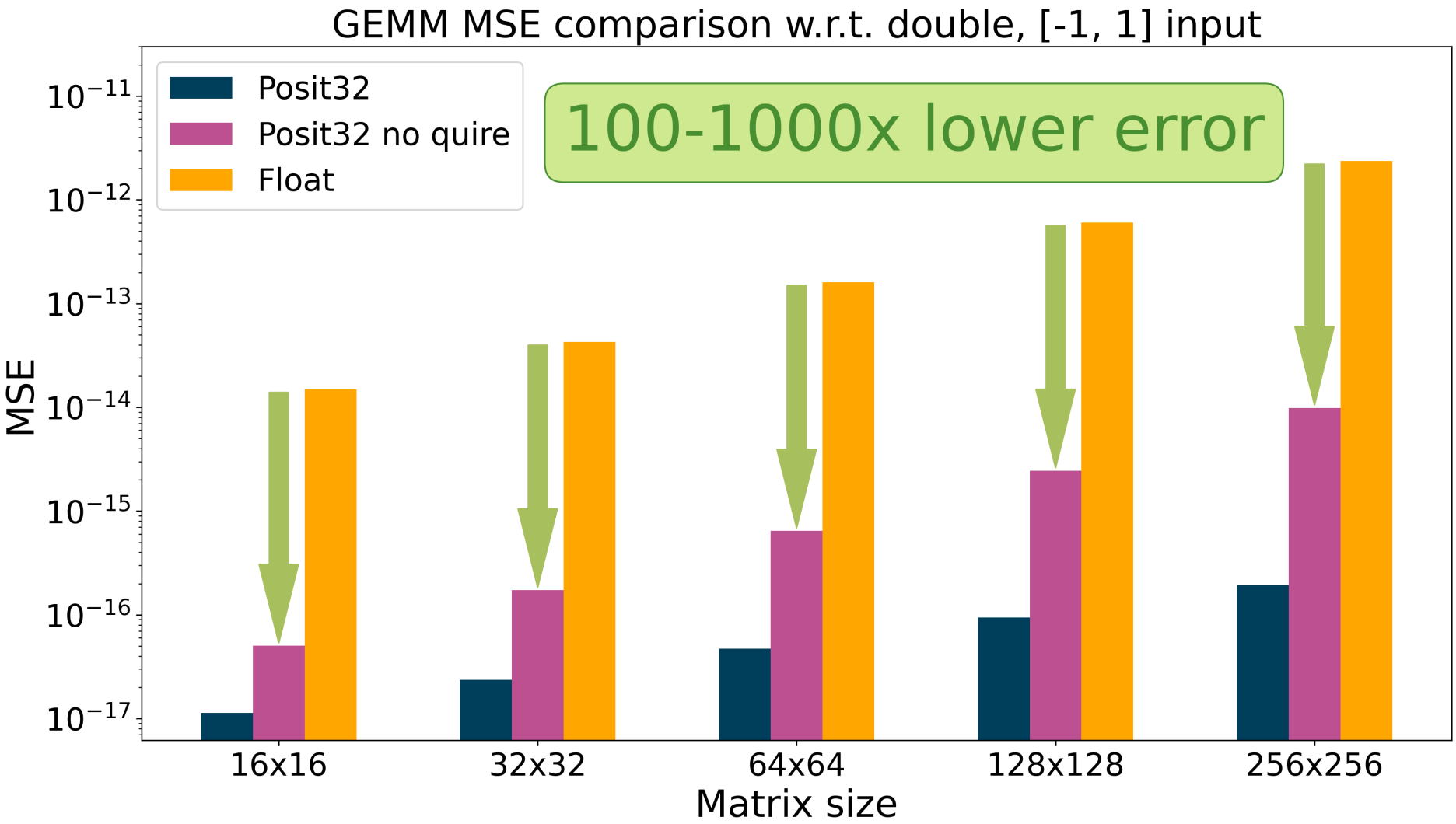
GEMM Accuracy Results



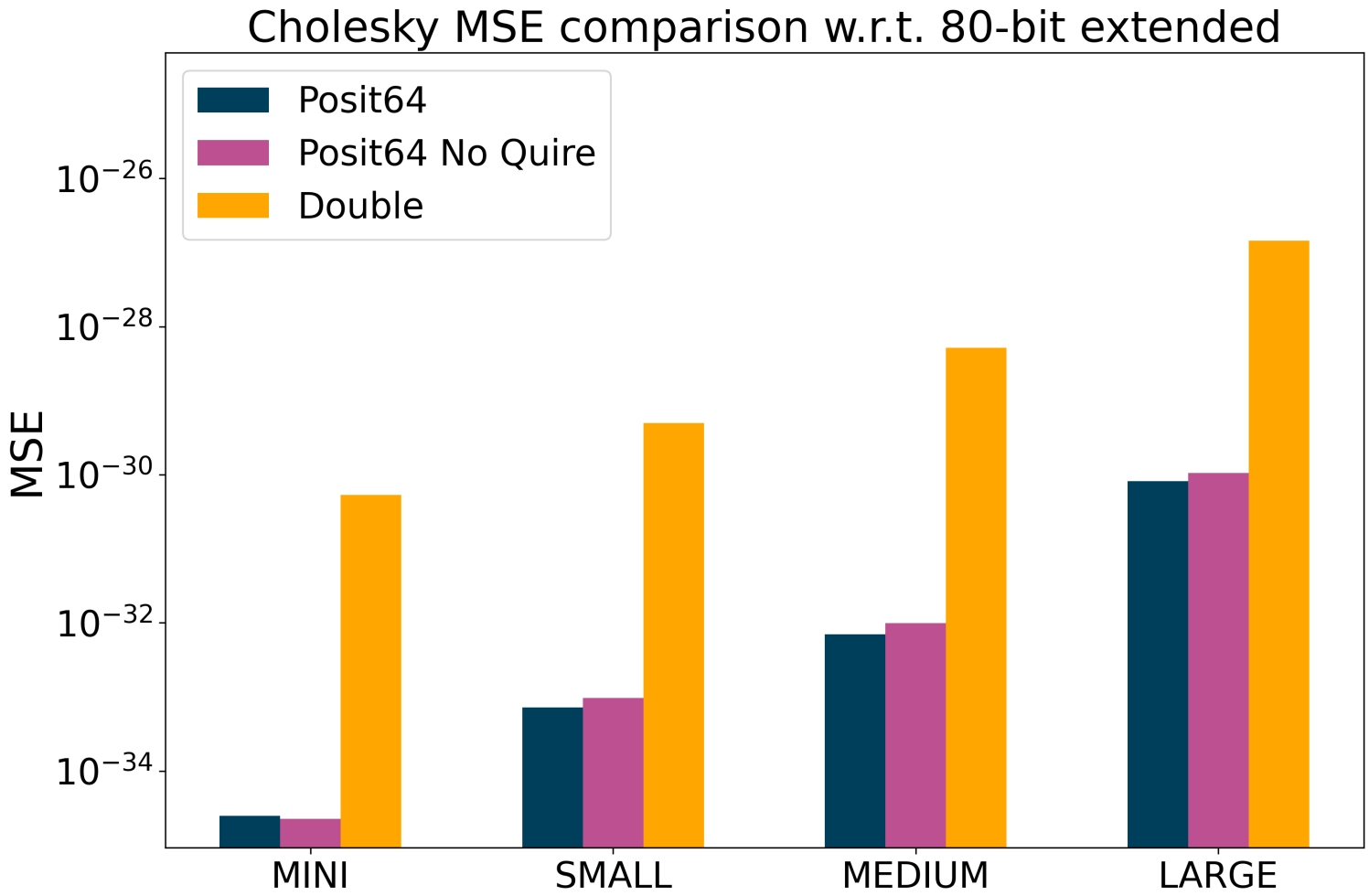
GEMM Accuracy Results



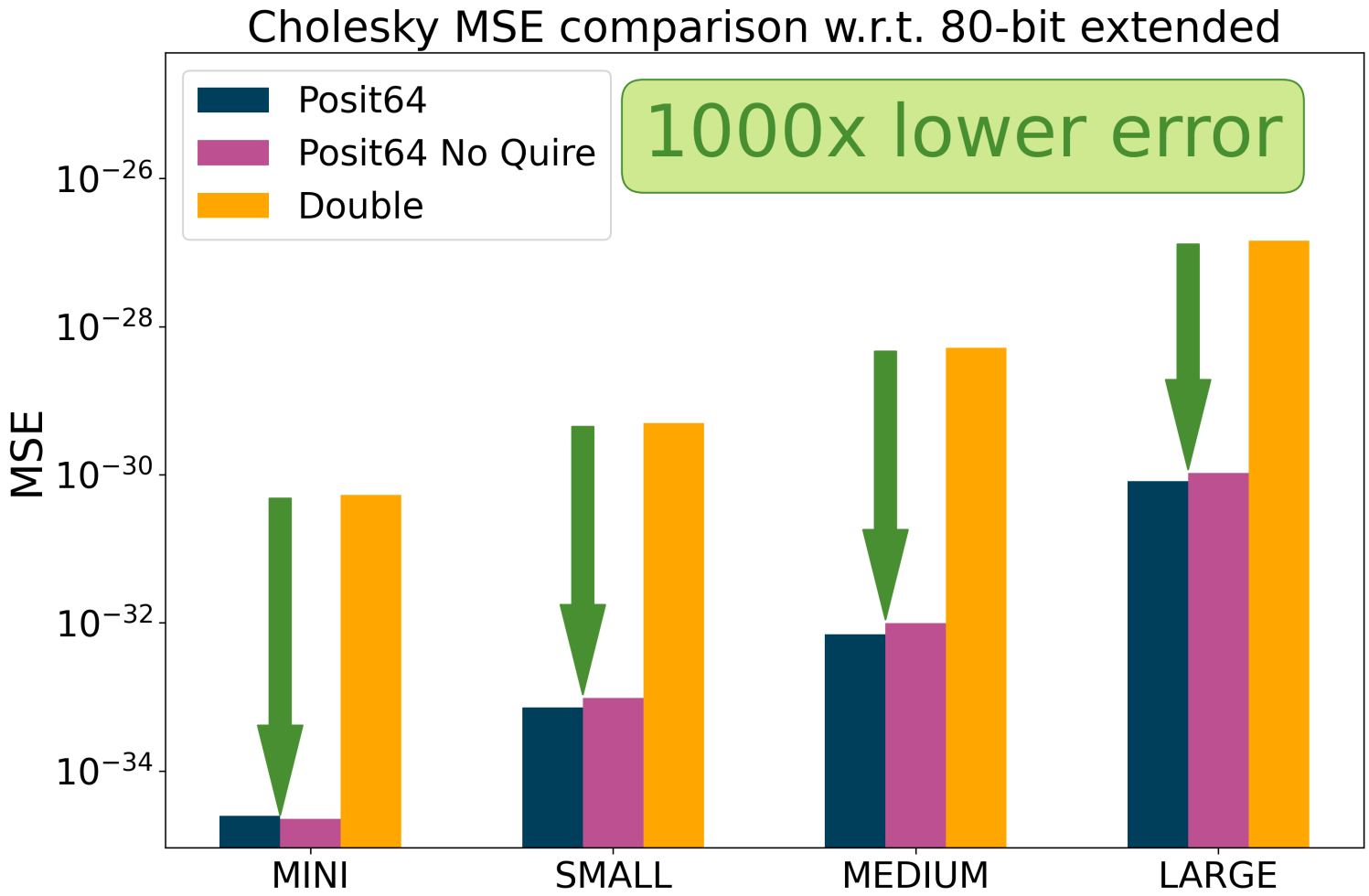
GEMM Accuracy Results



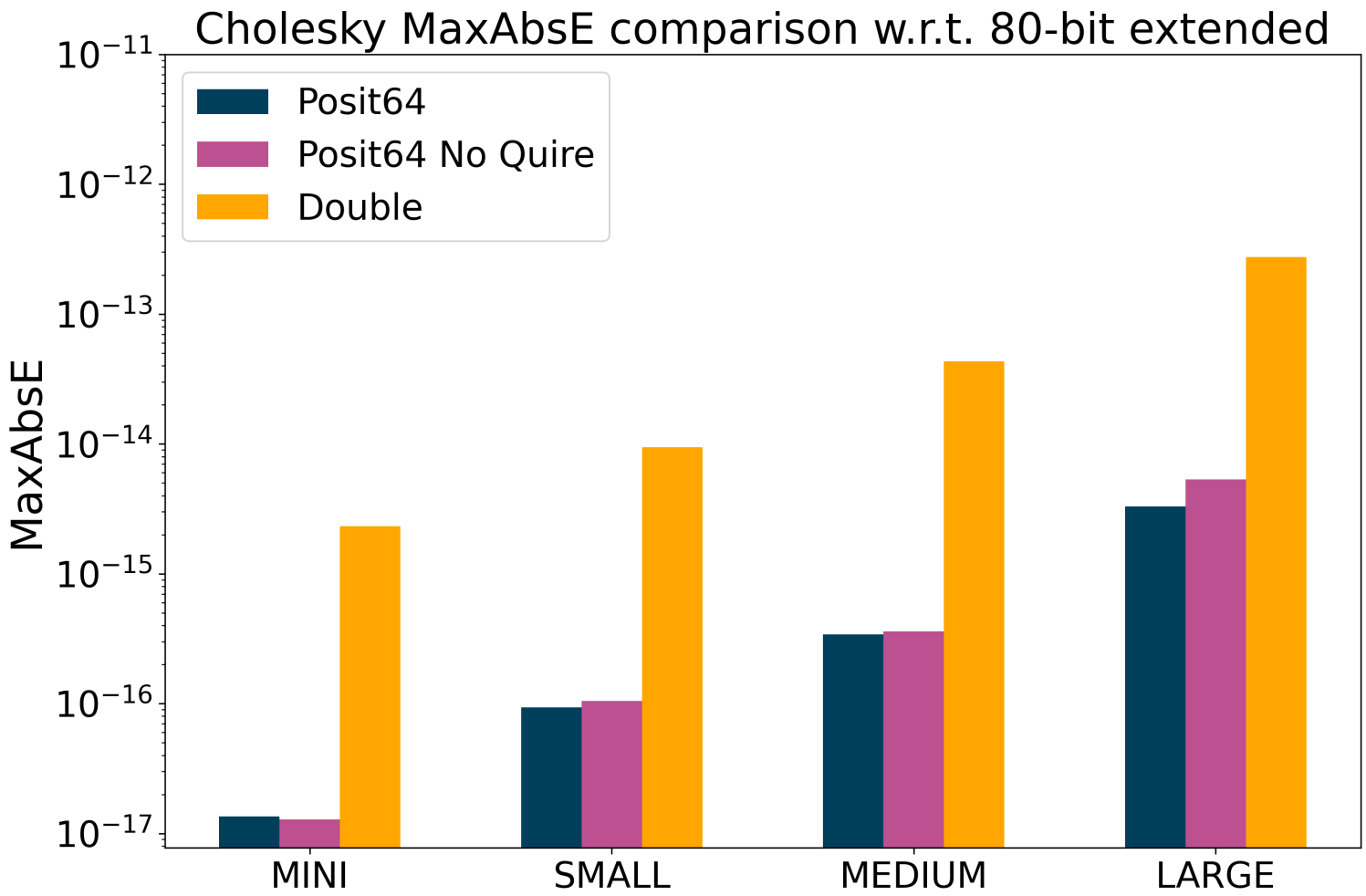
Cholesky Accuracy Results – Mean Squared Error



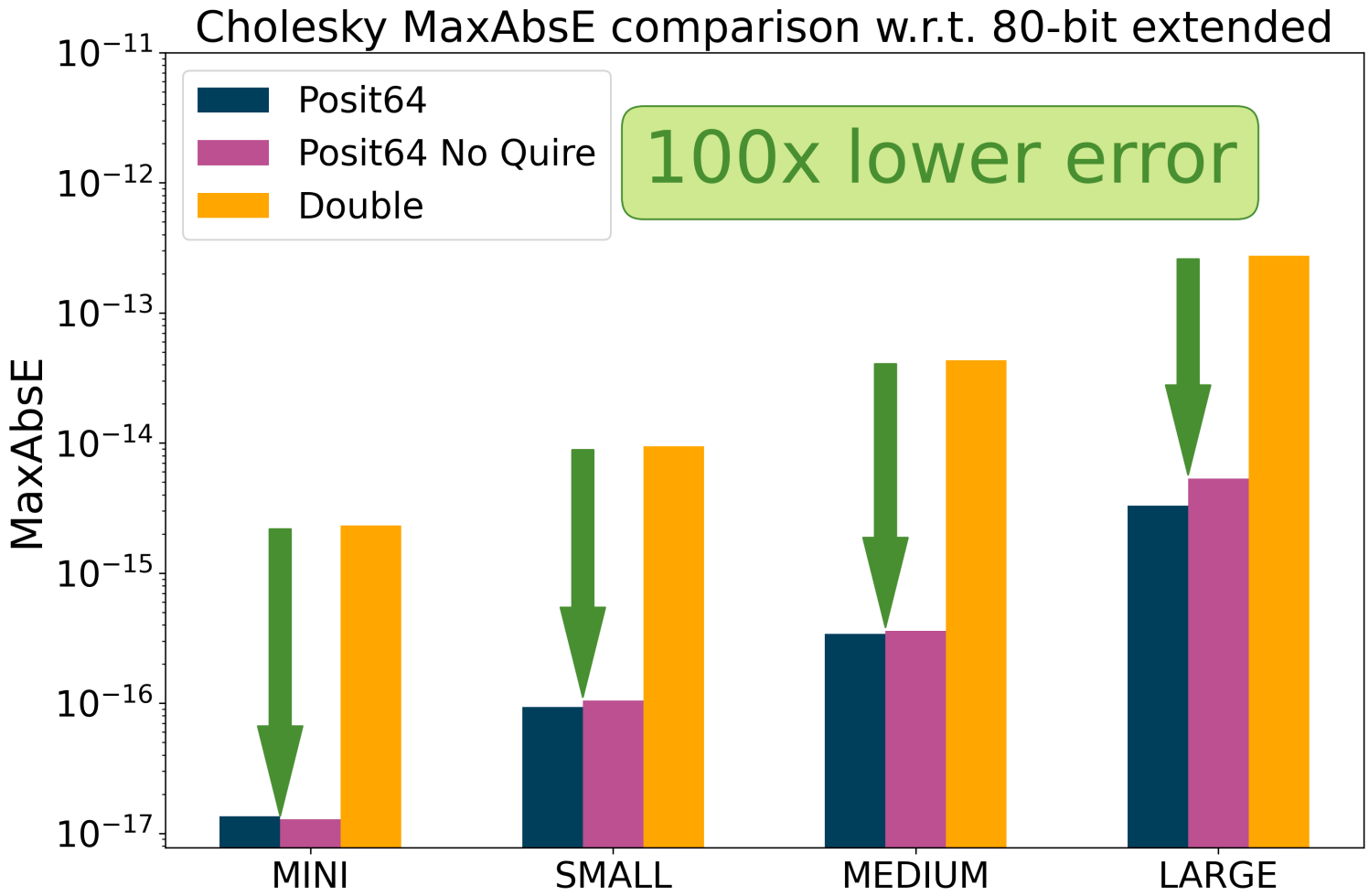
Cholesky Accuracy Results – Mean Squared Error



Cholesky Accuracy Results – Maximum Absolute Error



Cholesky Accuracy Results – Maximum Absolute Error



Performance Results

- GEMM timing results:

Matrix size	256×256
32-bit float	13.9 s
Posit32	13.9 s

As fast as floats!

Performance Results

- GEMM timing results:

Matrix size	256 × 256
32-bit float	13.9 s
Posit32	13.9 s
32-bit float no FMADD	15.0 s
Posit32 no quire	15.0 s

As fast as floats!

Performance Results

- GEMM timing results:

Matrix size	256 × 256
32-bit float	13.9 s
Posit32	13.9 s
32-bit float no FMADD	15.0 s
Posit32 no quire	15.0 s

As fast as floats!

- Cholesky timing results:

Data size		LARGE
Cholesky	Double	870.03 s
	Posit64	767.29 s
	Posit64 no quire	1038.1 s

Conclusions

Hardware

- PERCIVAL $\rightarrow$ CVA6 RV64GC + **Xposit**
- FPGA synthesis results $\rightarrow$ room for improvement

Open hardware!

Conclusions

Hardware

- PERCIVAL $\rightarrow$ CVA6 RV64GC + **Xposit**
- FPGA synthesis results $\rightarrow$ room for improvement

Open hardware!

Software

- Xposit RISC-V custom extension included in LLVM
- Posits with quire can perform as fast as floats
- Up to 4 OoM lower MSE and 2 OoM lower MaxAbsE
- Alternative solution depending on the application needs

Also open source!

PERCIVAL: Integrating Posit and Quire Arithmetic into the RISC-V Ecosystem

David Mallasén, Raul Murillo, Alberto A. Del Barrio, Guillermo Botella, Luis Piñuel, Manuel Prieto-Matias

dmallasen@ucm.es

https://github.com/artecs-group/PERCIVAL

Department of Computer Architecture and Automation
Computer Science and Engineering Faculty
Complutense University of Madrid

UNIVERSIDAD COMPLUTENSE MADRID

Backup slides

Posit Arithmetic: Quire Accumulator Register

- Fixed-point accumulator: for posit32 $\rightarrow$ 512-bit quire
- $2^{31} - 1$ fused MAC operations without accuracy loss
- Workflow of operations:
  - `qclr` : $quire \leftarrow 0$
  - `qmadd` : $quire \leftarrow quire + (posit_1 \times posit_2)$
  - `qround` : $posit \leftarrow \mathsf{round}(quire)$

Dot product: $\mathbf{a} \cdot \mathbf{b} = a_1 b_1 \quad + a_2 b_2 \quad + \cdots + a_n b_n$

Matrix multiply: $c_{ij} = a_{i1} b_{1j} + a_{i2} b_{2j} + \cdots + a_{in} b_{nj}$

Xposit RISC-V Custom Extension

31 27	26 25	24 20	19 15	14 12	11 7	6 0	
imm[11:0]			rs1	001	rd	0001011	PLW
imm[11:0]			rs1	101	rd	0001011	PLD
imm[11:5]		rs2	rs1	011	imm[4:0]	0001011	PSW
imm[11:5]		rs2	rs1	110	imm[4:0]	0001011	PSD

Load/Store

00000	10	rs2	rs1	000	rd	0001011	PADD.S
00001	10	rs2	rs1	000	rd	0001011	PSUB.S
00010	10	rs2	rs1	000	rd	0001011	PMUL.S
00011	10	rs2	rs1	000	rd	0001011	PDIV.S
00100	10	rs2	rs1	000	rd	0001011	PMIN.S
00101	10	rs2	rs1	000	rd	0001011	PMAX.S
00110	10	00000	rs1	000	rd	0001011	PSQRT.S

Computational

00111	10	rs2	rs1	000	00000	0001011	QMADD.S
01000	10	rs2	rs1	000	00000	0001011	QMSUB.S
01001	10	00000	00000	000	00000	0001011	QCLR.S
01010	10	00000	00000	000	00000	0001011	QNEG.S
01011	10	00000	00000	000	rd	0001011	QROUND.S

Quire

01100	10	00000	rs1	000	rd	0001011	PCVT.W.S
01101	10	00000	rs1	000	rd	0001011	PCVT.WU.S
01110	10	00000	rs1	000	rd	0001011	PCVT.L.S
01111	10	00000	rs1	000	rd	0001011	PCVT.LU.S
10000	10	00000	rs1	000	rd	0001011	PCVT.S.W
10001	10	00000	rs1	000	rd	0001011	PCVT.S.WU
10010	10	00000	rs1	000	rd	0001011	PCVT.S.L
10011	10	00000	rs1	000	rd	0001011	PCVT.S.LU

Conversion Posit-Integer

10100	10	rs2	rs1	000	rd	0001011	PSGNJ.S
10101	10	rs2	rs1	000	rd	0001011	PSGNJN.S
10110	10	rs2	rs1	000	rd	0001011	PSGNJX.S
10111	10	00000	rs1	000	rd	0001011	PMV.X.W
11000	10	00000	rs1	000	rd	0001011	PMV.W.X

Register move

11001	10	rs2	rs1	000	rd	0001011	PEQ.S
11010	10	rs2	rs1	000	rd	0001011	PLT.S
11011	10	rs2	rs1	000	rd	0001011	PLE.S

Comparison

ASIC Synthesis Results

- TSMC 45nm at 200MHz
- 32-bit FPU: 30691μm^2, 27.26 mW
- 32-bit posit with quire:
  - ~2.5x more area and power than FP
- 32-bit posit without quire:
  - 32% more area and 38% more power

Name	Area (μm^2)	Power (mW)
PAU top	13462.15	12.69
Posit Add	4075.31	3.59
Posit Mult	8635.37	9.98
Posit ADiv	2540.87	2.41
Posit ASqrt	1722.84	1.61
Quire MAC	30419.12	26.07
Quire to Posit	6026.76	4.04
Int to Posit	905.99	0.68
Long to Posit	1423.43	0.96
UInt to Posit	869.77	0.66
ULong to Posit	1353.11	0.94
Posit to Int	966.67	0.71
Posit to Long	1810.33	1.38
Posit to UInt	958.44	0.68
Posit to ULong	1800.22	1.33
PAU total	76970.38	67.73
PAU w/o quire	40524.62	37.62

GEMM Benchmark with Xposit

Require: Posit matrices `a` and `b` of size `n`×`n`.
Ensure: Posit matrix `c` = `ab`.

```
for i = 0 to n-1 do
  for j = 0 to n-1 do
    asm(”qclr.s”:::); {Clear the quire}
    for k = 0 to n-1 do
      asm
        ”plw      pt0,0(%0)” {Load posit a and b}
        ”plw      pt1,0(%1)”
        ”qmadd.s pt0,pt1” {Accumulate on the quire}
        :: ”r” (&a[i * n + k]), ”r” (&b[k * n + j]):
      end asm
    end for
    asm
      ”qround.s pt2” {Round the quire to a posit}
      ”psw      pt2,0(%1)” {Store the result in c}
      : ”=rm” (c[i * n + j]) : ”r” (&c[i * n + j]) :
    end asm
  end for
end for
```