Hardware and Software Support for Mixed Precision Computing: a Roadmap for Embedded and HPC Systems

High Performance Computing Summer School, 12-16 June, Pavia, Italy

William Fornaciari[1,2]

[1] *DEIB - Politecnico di Milano*

[2] *CINI National Laboratory HPC-KTT*

POLITECNICO
MILANO 1863

DIPARTIMENTO DI ELETTRONICA
INFORMAZIONE E BIOINGEGNERIA

HEAP LAB

textarossa

APROPOS

Approximate and mixed-precision computing

Approximate Computing

- Trade off computation accuracy for performance and energy
- Variety of approaches: voltage scaling, loop perforation, precision scaling, etc.

Mixed-Precision Computing

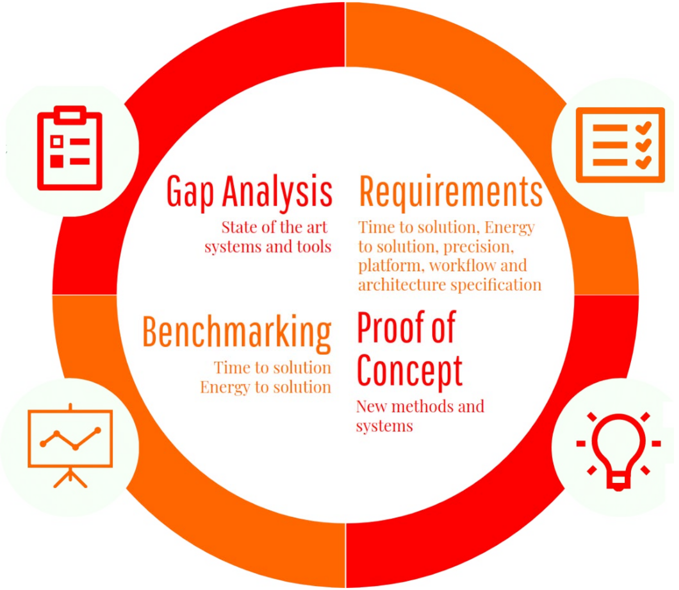
- Fine-grained control of accuracy-performance/energy trade-off
- Modifying the data types involved in computation
- Error-prone when done manually, requires automation

The TEXTAROSSA EuroHPC project

Co-design approach to heterogeneous HPC solutions

Key objectives:

- energy efficiency and thermal control
- extreme computation efficiency via HW accelerators and new arithmetics
- seamless integration of reconfigurable accelerators in heterogeneous multi-node platforms


Gap Analysis
State of the art systems and tools
Requirements
Time to solution, Energy to solution, precision, platform, workflow and architecture specification
Benchmarking
Time to solution Energy to solution
Proof of Concept
New methods and systems

The APROPOS MSCA-ITN project

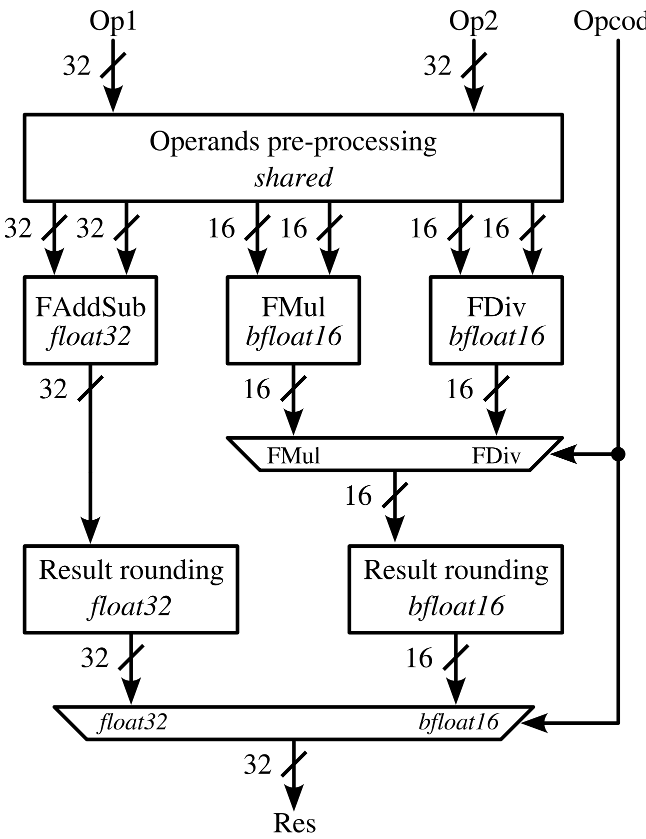
Approximate Computing for Power and Energy Optimisation

Optimization through energy-accuracy trade-offs

Decrease energy consumption:

- Distributed computing
- Communications for cloud-based cyber-physical systems
- Embedded systems

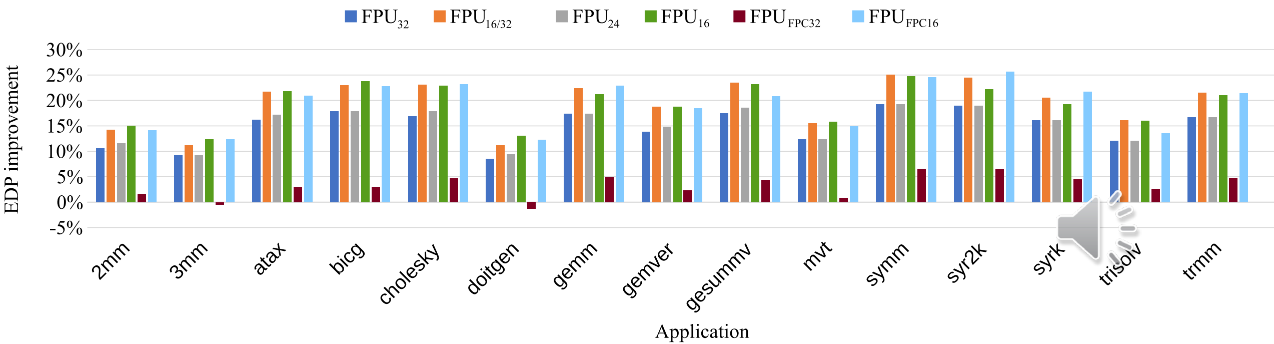
Mixed-precision floating-point unit



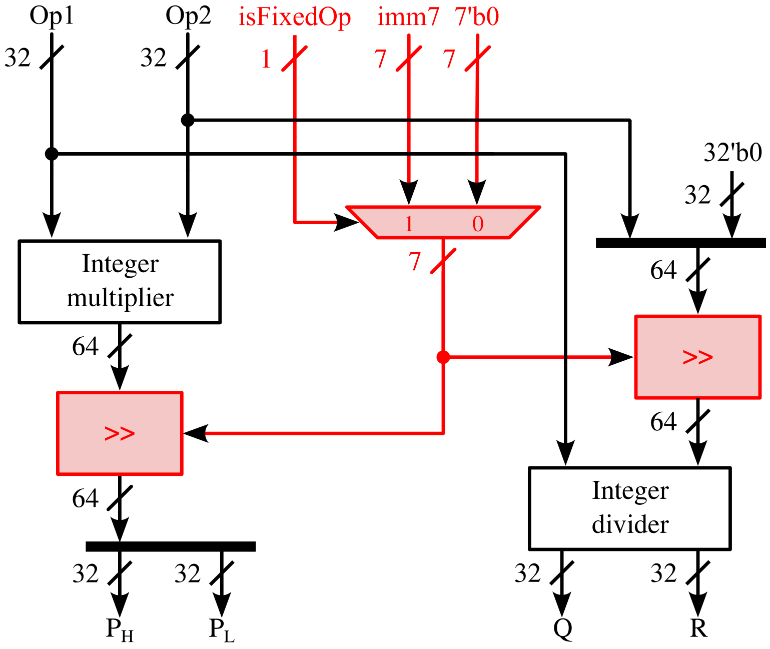
Operation precision configurable at design time depending on 1) target applications, 2) accuracy requirements, 3) resource utilization constraints

No compiler-side changes required

Up to -19% EDP and -21% area compared to state-of-the-art, with <3% avg accuracy error



Mixed-precision fixed-point hardware support

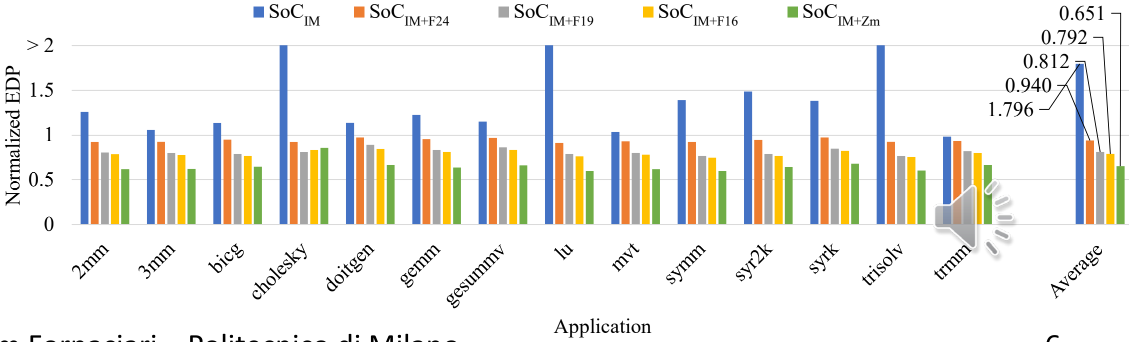


Supports at run time any 32-bit fixed-point format encoded in fixed-point mul/div opcode

Reuses hardware of integer mul/div (RISC-V M extension) with limited changes

+4% area vs. RISC-V IM CPU

-35% EDP vs. float exec. on RISC-V IMF CPU



Precision-tuning support at compiler level

Lower level of granularity = more opportunities for optimizations

Access to information about the specifics of target hardware

Non-intrusive and transparent for the programmer

Benefits from highly developed compiler ecosystem

Precision-tuning tools: requirements

Optimize programs containing loops, conditionals and memory operations

Work with programs written in commonly-used programming language

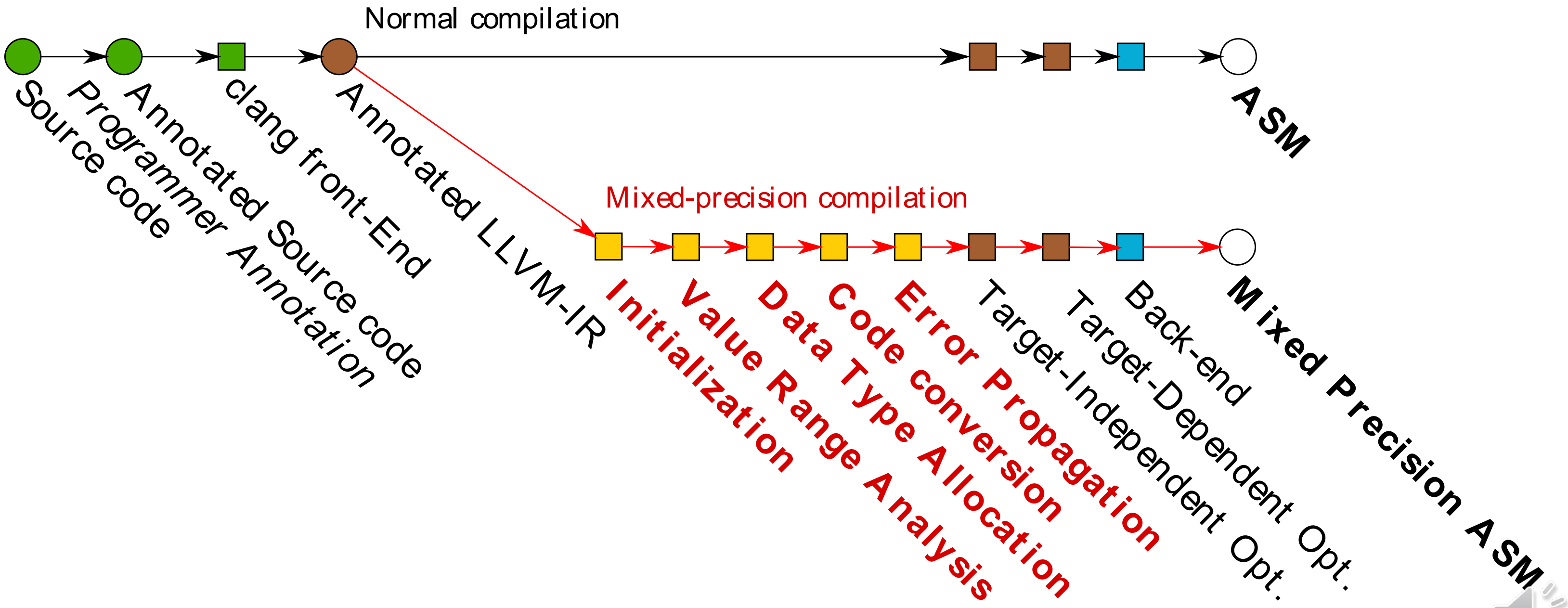
Support wide variety of execution platforms

Work with a modern compilation ecosystem

Compiler-based precision-tuning tools

Tool	Validation	Input Language	Algorithm
TAFFO	Static	C, C++	Interval Arithmetic, Affine Arithmetic, ILP
Rosa	Static	Scala	Interval Arithmetic, Affine Arithmetic, SMT
Daisy	Static	Scala, C	Interval Arithmetic, Affine Arithmetic, SMT, rewriting rules
FloatSmith	Dynamic	C, C++	Algorithmic Differentiation, Delta-Debugging, Hierarchical Composition, Genetic Search
Precimonious	Dynamic	C, C++	Delta-Debugging

Precision-tuning compiler: architecture



Precision-tuning compiler: architecture

1. Initializer

   Code preprocessing

2. Value Range Analysis

   Propagates value ranges to each intermediate LLVM-IR instruction

3. Data Type Allocation

   Decides data types and bit partitioning (for fixed point)

4. Code conversion

   Actual code manipulation

5. Feedback Estimation

   Estimates error (upper bound)

   Estimates speedup (faster / slower / similar)

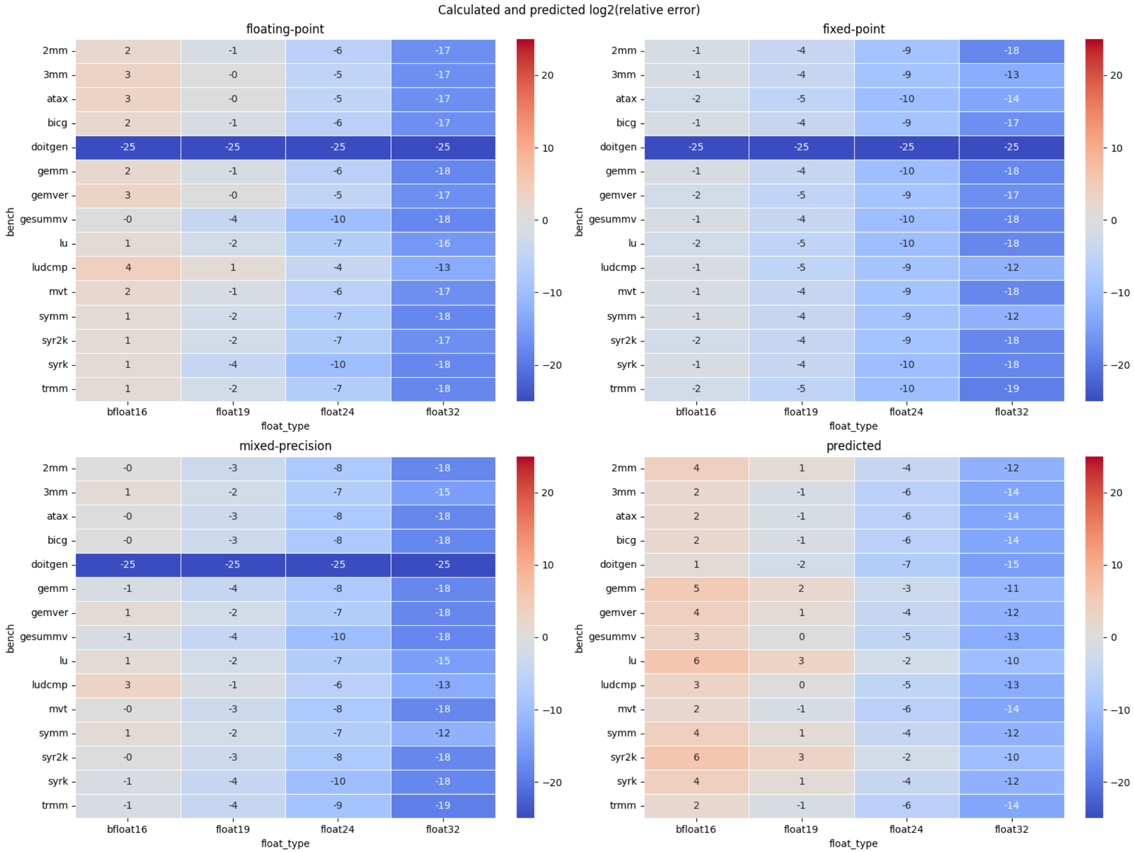
Hardware-compiler synergy

Mixed-precision is gaining support in the research and industry

- New hardware extensions are being designed, including support for new data types such as *bfloat16*
- Compiler-based tools developed to support the programmer in selecting the best solution for their applications

Need to combine these developments in a hardware/software co-design approach to maximize benefits

Hardware-software co-design



Calculated and predicted log2(relative error)

floating-point

bench	bfloat16	float19	float24	float32
2mm	2	-1	-6	-17
3mm	3	-0	-5	-17
atax	3	-0	-5	-17
bicg	2	-1	-6	-17
doitgen	-25	-25	-25	-25
gemm	2	-1	-6	-18
gemver	3	-0	-5	-17
gesummv	-0	-4	-10	-18
lu	1	-2	-7	-16
ludcmp	4	1	-4	-13
mvt	2	-1	-6	-17
symm	1	-2	-7	-18
syr2k	1	-2	-7	-17
syrk	1	-4	-10	-18
trmm	1	-2	-7	-18

fixed-point

bench	bfloat16	float19	float24	float32
2mm	-1	-4	-9	-18
3mm	-1	-4	-9	-13
atax	-2	-5	-10	-14
bicg	-1	-4	-9	-17
doitgen	-25	-25	-25	-25
gemm	-1	-4	-10	-18
gemver	-2	-5	-9	-17
gesummv	-1	-4	-10	-18
lu	-2	-5	-10	-18
ludcmp	-1	-5	-9	-12
mvt	-1	-4	-9	-18
symm	-1	-4	-9	-12
syr2k	-2	-4	-9	-18
syrk	-1	-4	-10	-18
trmm	-2	-5	-10	-19

mixed-precision

bench	bfloat16	float19	float24	float32
2mm	-0	-3	-8	-18
3mm	1	-2	-7	-15
atax	-0	-3	-8	-18
bicg	-0	-3	-8	-18
doitgen	-25	-25	-25	-25
gemm	-1	-4	-8	-18
gemver	1	-2	-7	-18
gesummv	-1	-4	-10	-18
lu	1	-2	-7	-15
ludcmp	3	-1	-6	-13
mvt	-0	-3	-8	-18
symm	1	-2	-7	-12
syr2k	-0	-3	-8	-18
syrk	-1	-4	-10	-18
trmm	-1	-4	-9	-19

predicted

bench	bfloat16	float19	float24	float32
2mm	4	1	-4	-12
3mm	2	-1	-6	-14
atax	2	-1	-6	-14
bicg	2	-1	-6	-14
doitgen	1	-2	-7	-15
gemm	5	2	-3	-11
gemver	4	1	-4	-12
gesummv	3	0	-5	-13
lu	6	3	-2	-10
ludcmp	3	0	-5	-13
mvt	2	-1	-6	-14
symm	4	1	-4	-12
syr2k	6	3	-2	-10
syrk	4	1	-4	-12
trmm	2	-1	-6	-14

Compiler: different precision mixes

Hardware: different sizes of FP-unit

Select optimal HW implementation based on program and error/ performance trade-off

Future works

Hardware/software co-design approach to precision tuning

- Optimize program for the best mix of data types depending on the architectural options available
- Output configuration for generation/selection of the hardware platform

TAFFO support for Heterogeneous Parallel HPC Architectures

- OpenMP support (done)
- OpenCL support
- CUDA support