MiniFloat-NN: A RISC-V ISA Extension for Low-Precision NN Training

Luca Bertaccini (ETH Zurich)

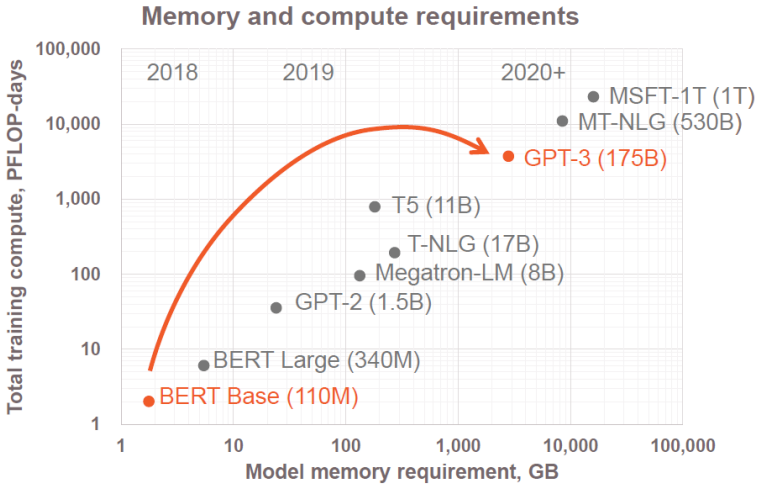
@pulp_platform

pulp-platform.org

youtube.com/pulp_platform

THE LINUX FOUNDATION

The Rapid Growth of AI



S. Lie, "Thinking outside the die: Architecting the ML accelerator of the future"

- NN models' memory and compute requirements are rapidly growing
- Technology scaling is not sufficient

- Required algorithmic and architectural advancements

Algorithmic Advancements

- New low-precision data types:
  - 32-bit → 19-bit → 16-bit → 8-bit floating-point (FP) data types

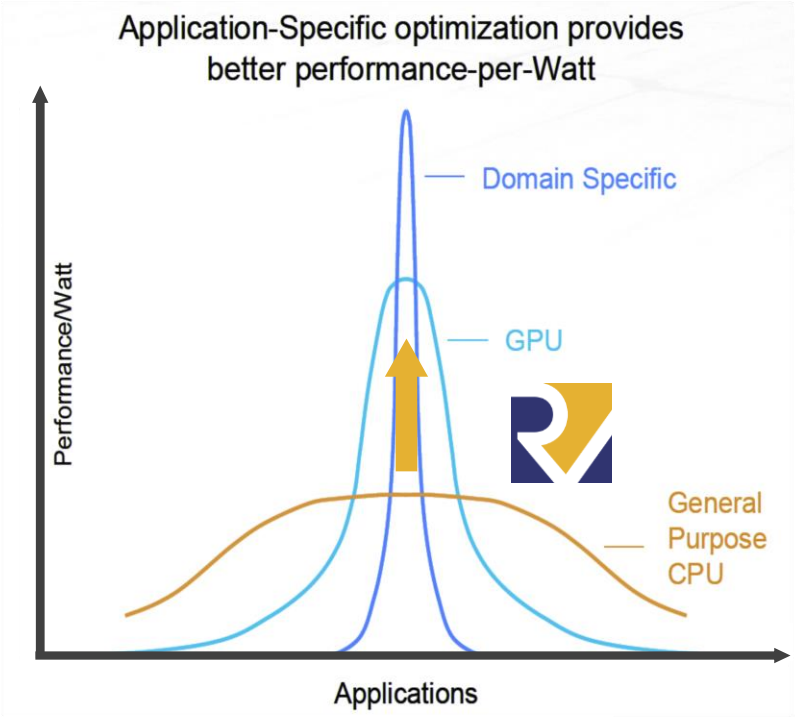
- Lower memory requirements
- Opportunities for more efficient hardware architectures
- Wide interest for standardization (RISC-V FP SIG, IEEE P3109)

- New **mixed** and **low-precision training** algorithms have been developed to exploit the resilience of NN models to noise
- Expanding/Widening **operations** in which the accumulation is performed in higher precision

Application-Specific Optimizations

- The more an architecture is specialized, the narrower the application scenarios in which it represents an attractive solution

- General-purpose **RISC-V** CPUs can be extended with domain-specific ISA extension:
  - Opportunities for deeply optimizating the efficiency of the processing element in a large manycore system
  - Without compromising the baseline standardized ISA



[AMD Naffziger ISCAS22]

Building an HPC RV System Extended for Efficient Training

MiniFloat-NN: a RISC-V ISA extension for low-precision NN training

Efficient clusters to hierarchically replicate to build HPC RV manycore systems

Set of **instructions** and **formats** to efficiently compute NN workloads

Hardware functional **unit** and **system** integration

Building an HPC RV System Extended for Efficient Training

MiniFloat-NN: a RISC-V ISA extension for low-precision NN training

Efficient clusters to hierarchically replicate to build HPC RV manycore systems

Set of **instructions** and **formats** to efficiently compute NN workloads

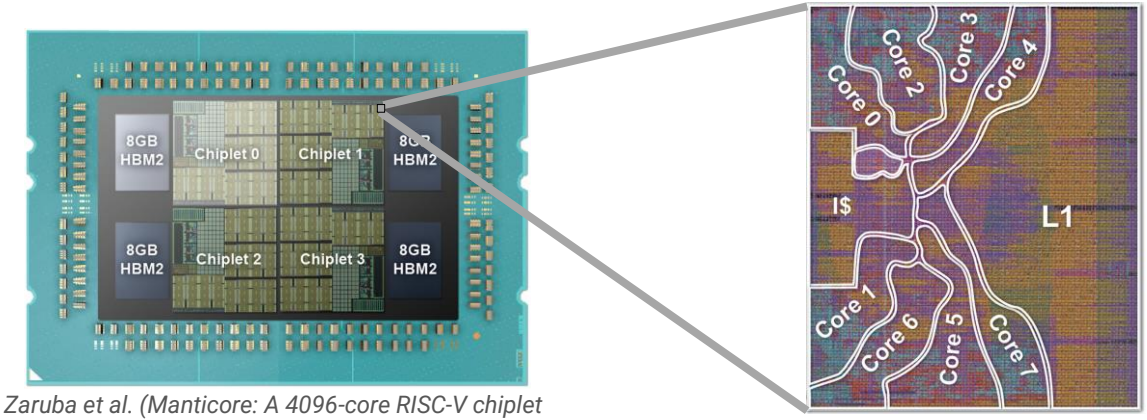
Hardware functional **unit** and **system** integration

ETH zürich

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

THE LINUX FOUNDATION

Large Manycore RISC-V Architecture - Manticore

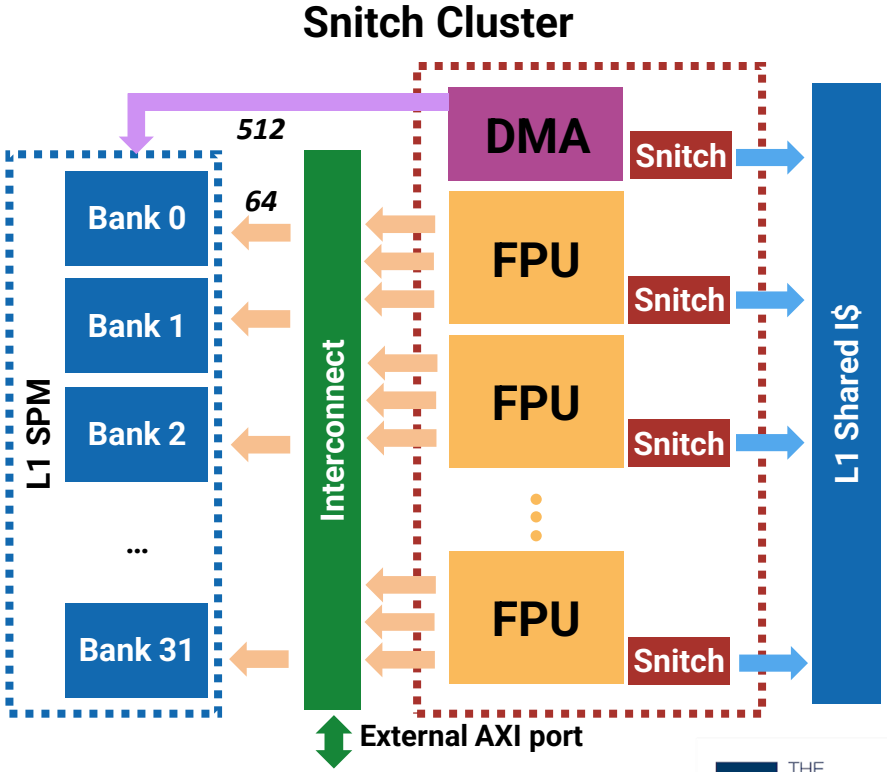


Zaruba et al. (Manticore: A 4096-core RISC-V chiplet architecture for ultraefficient floating-point computing)

- Manticore: a chiplet-based hierarchically-scalable architecture
- Linux-capable host + large manycore accelerator built upon the replication of efficient L1-shared **compute clusters**

Efficient RISC-V Compute Clusters: Scalar + Parallel

- Manticore cluster: **Snitch** compute cluster*
- **Small** integer **scalar** 32b **cores** coupled with **large SIMD FPUs** (FP64, FP32) sharing a scratchpad memory
- **ISA extensions** that implicity encode loads/stores to register reads/writes + loops handled in HW → **~90% FPU utilization**
- Need for narrow FP formats and expanding instructions to efficiently tackle mixed, low-precision NN training



*https://github.com/pulp-platform/snitch

Building an HPC RV System Extended for Efficient Training

MiniFloat-NN: a RISC-V ISA extension for low-precision NN training

Efficient clusters to hierarchically replicate to build HPC RV manycore systems

Set of **instructions** and **formats** to efficiently compute NN workloads

Hardware functional **unit** and **system** integration

Without Mixed-Precision Capabilities

In the case of **no mixed-precision** support:

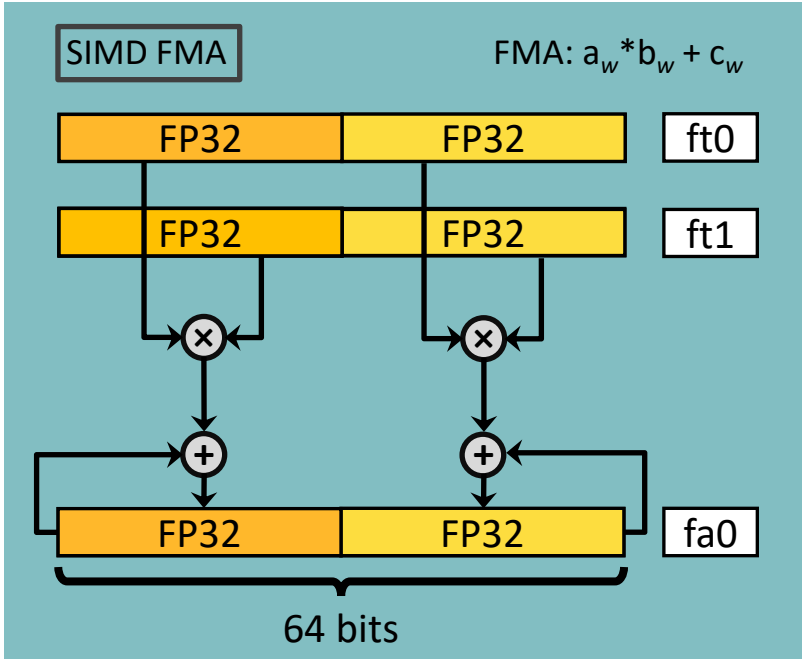
If the application can work on FP16/FP16ALT inputs but accumulation in FP32 required:

- FP16 with accumulation on FP32 requires additional conversion which affects the performance
- An FP32 kernel can be used

```
   <configure loop>
Loop:
   simd_fmacc.s fa0, ft0, ft1 } N
EndLoop:
   <reduction>
```

Without Mixed-Precision Capabilities

- SIMD FP32 FMA



In the case of **no mixed-precision** support:

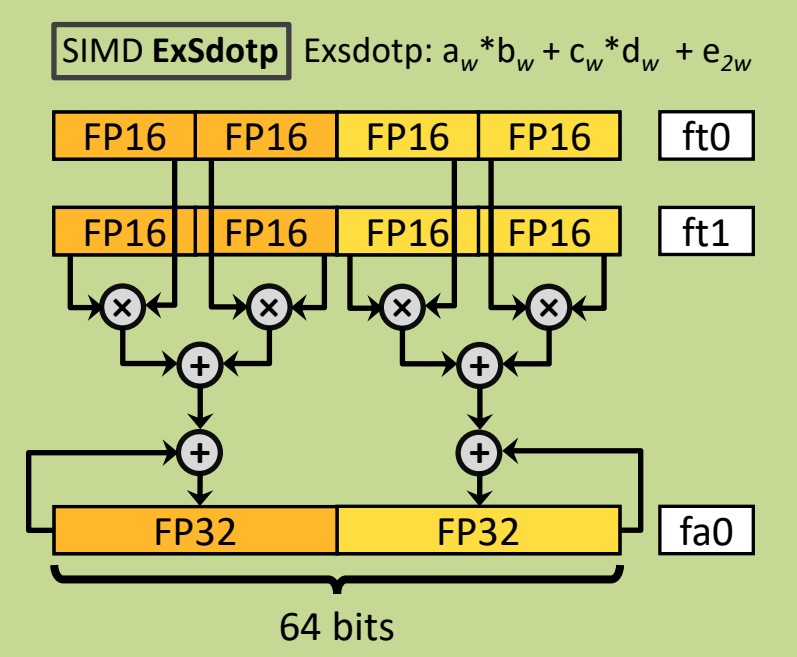
If the application can work on FP16/FP16ALT inputs but accumulation in FP32 required:

- FP16 with accumulation on FP32 requires additional conversion which affects the performance
- An FP32 kernel can be used

```
    <configure loop>
Loop:
    simd_fmacc.s fa0, ft0, ft1   } N
EndLoop:
    <reduction>
```

SIMD Expanding Sum of Dot Product for Mixed Precision

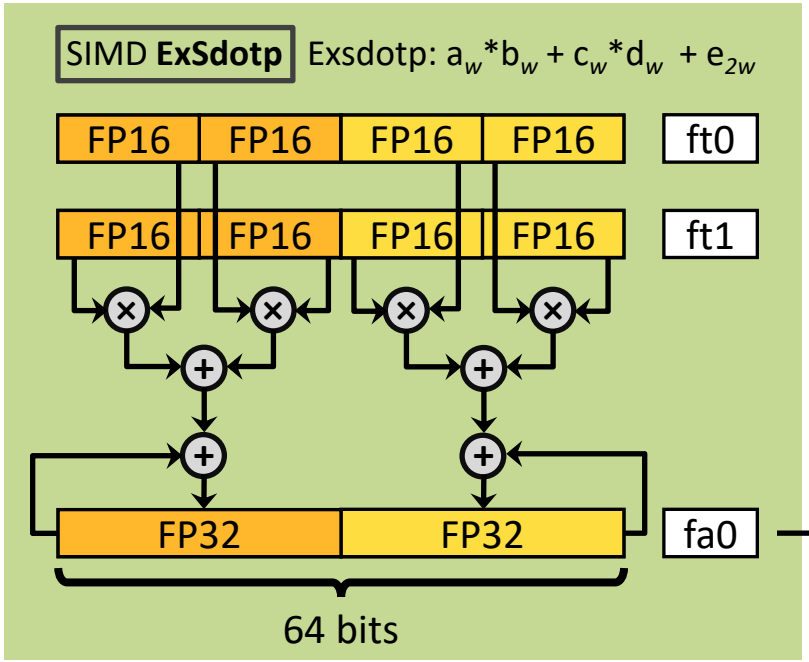
- With SIMD ExSdotp, 2x FLOP per cycle
- Same perf as SIMD FP16 FMA but more accurate



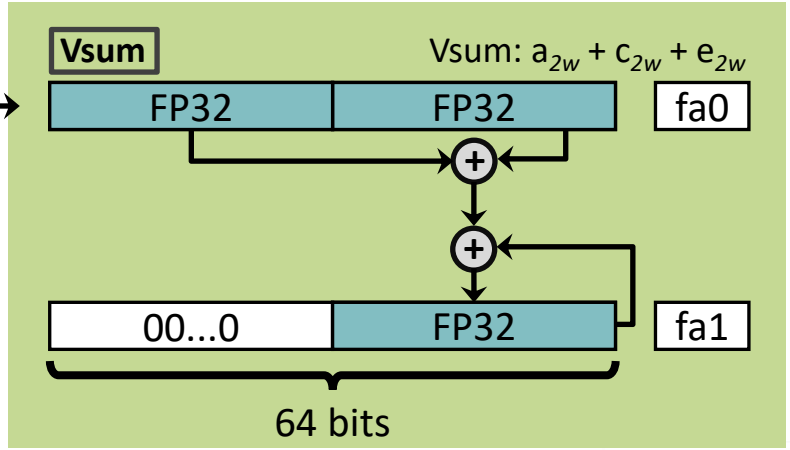

```
    <configure loop>
Loop:
    exsdotp.s.h fa0, ft0, ft1   } N/2
EndLoop:
    vsum.s      fa1, fa0 #reduction
```

SIMD Expanding Sum of Dot Product for Mixed Precision

- With SIMD ExSdotp, 2x FLOP per cycle
- Same perf as SIMD FP16 FMA but more accurate

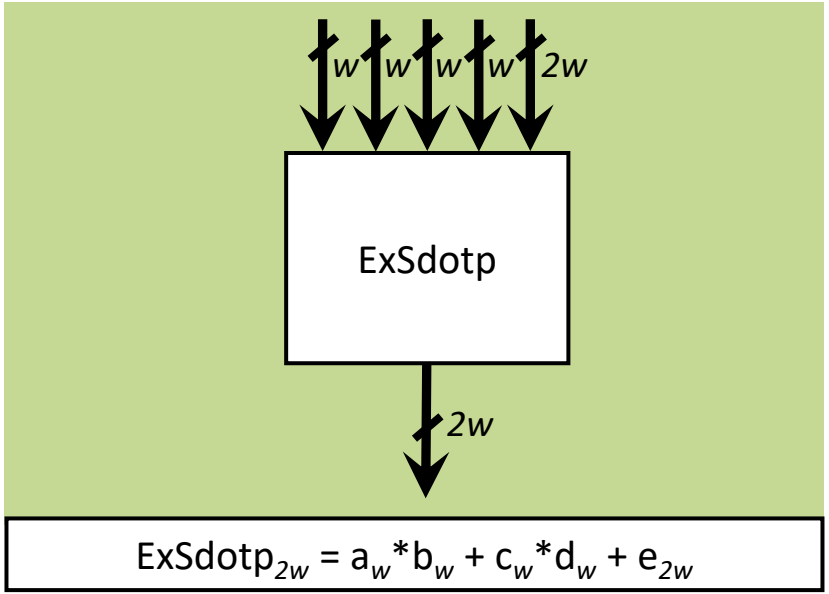



```
 <configure loop>
Loop:
   exsdotp.s.h fa0, ft0, ft1   } N/2
EndLoop:
   vsum.s      fa1, fa0 #reduction
```

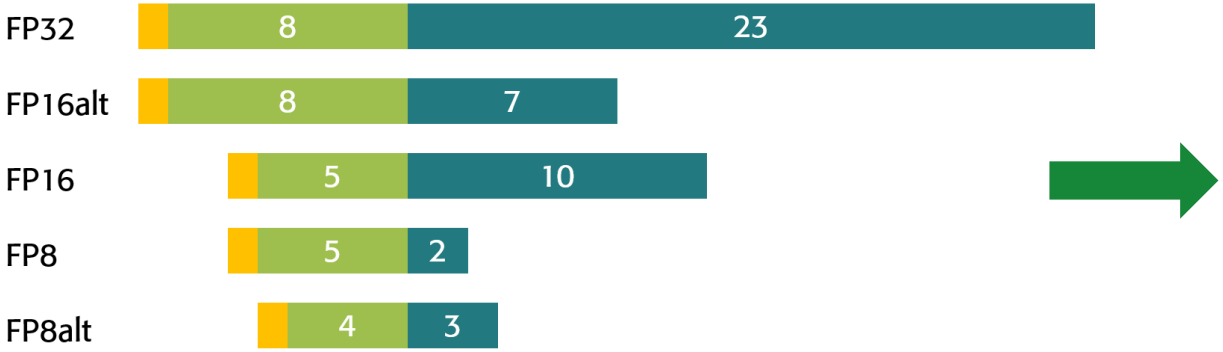


ExSdotp Offers Further Benefits

- Fused ExSdotp unit
- Single normalization and rounding step
- Opportunity to mitigate issues related to the non-associativity of the two consecutive additions



Targeted Floating-Point Formats



Many formats + mixed-precision would result in a large ISA extension

Alternate formats are enabled by FCSRs to reduce the number of instructions

- A parametric design to enable fast exploration of new FP formats
- ExSdotp source formats:
  - FP16alt (1, 8, 7)
  - FP16 (1, 5, 10)
  - FP8alt (1, 5, 2)
  - FP8 (1, 4, 3)
- ExSdotp destination formats:
  - FP32 (1, 8, 23)
  - FP16alt (1, 8, 7)
  - FP16 (1, 5, 10)

*Subnormals handled for all combinations of formats

Building an HPC RV System Extended for Efficient Training

MiniFloat-NN: a RISC-V ISA extension for low-precision NN training

Efficient clusters to hierarchically replicate to build HPC RV manycore systems

Set of **instructions** and **formats** to efficiently compute NN workloads

Hardware functional **unit** and **system** integration

ETH zürich

ALMA MATER STUDIORUM UNIVERSITÀ DI BOLOGNA

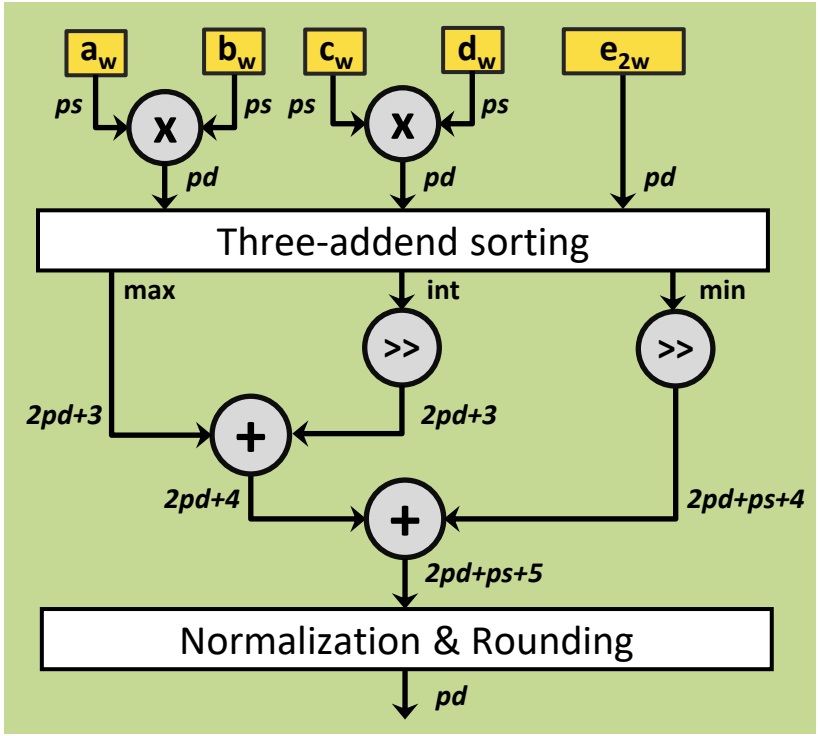
THE LINUX FOUNDATION

ExSdotp Unit

ExSdotp: an open-source* parameterized multi-format unit supporting ExSdotp and three-term additions (Vsum, ExVsum)

Expanding Sum of Dot Product Unit

- ***w*** *= source format bitwidth;* ***ps*** *= source format mant bits*
- ***2w*** *= destination format bitwidth;* ***pd*** *= destination format mant bits*
- ExSdotp $= a_w * b_w + c_w * d_w + e_{2w}$



Expanding Sum of Dot Product Unit

Issues arising in a single DOTP instruction (2 sums):

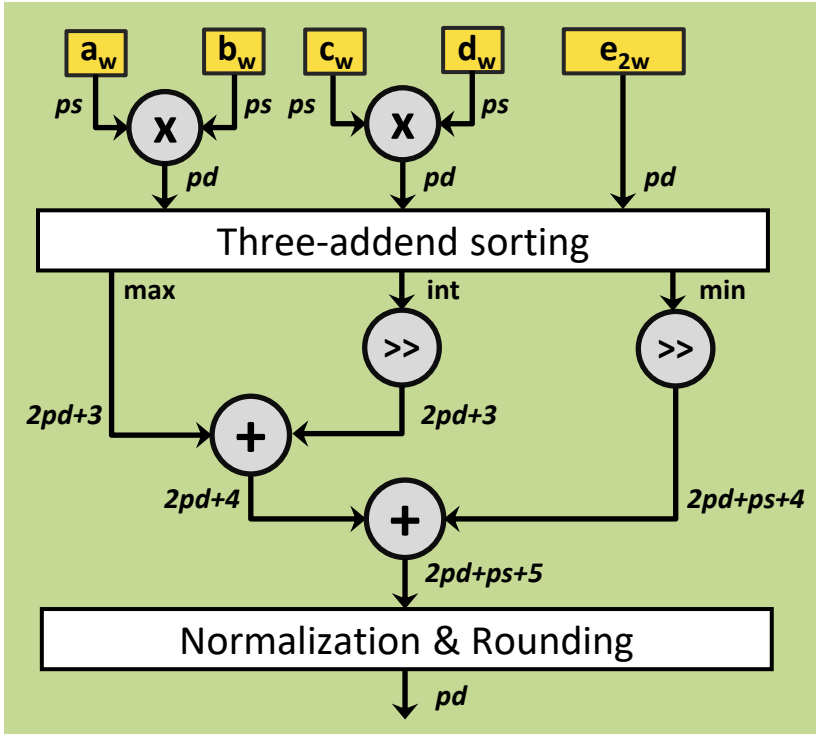
Cancellation: Three-addend sorting to prevent precision losses due to cancellation during the three-term addition:

- (a + b) – a might return 0 if a is much larger than b
- Decreasing order (abs value) -> (a – a) + b = b

Small quantities lost in the second addition: Gradually increasing the **internal bitwidth** to retain precision

Issues arising in long summations:

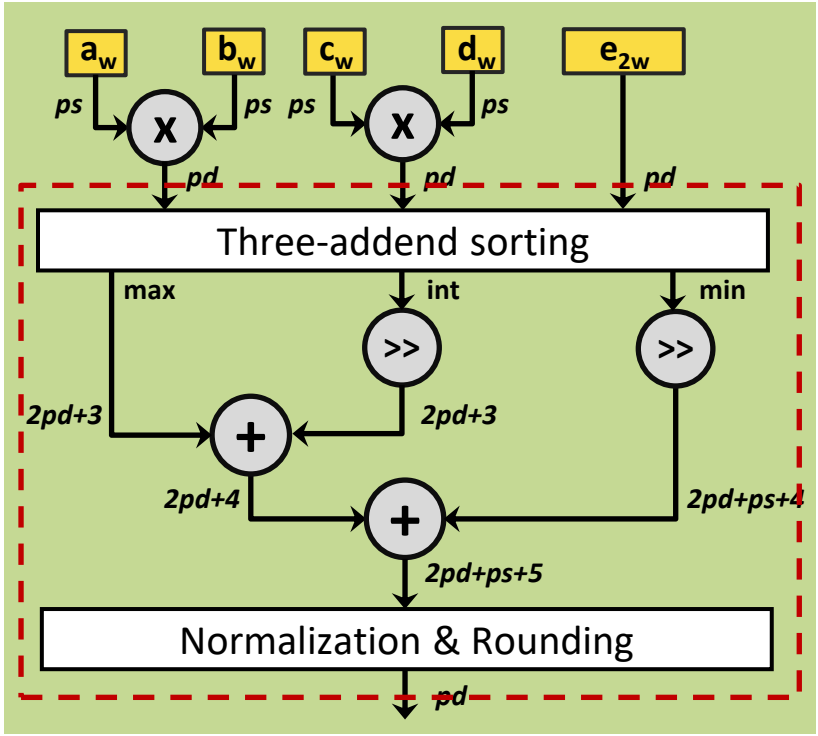
Stagnation: **Stochastic rounding** mitigates stagnation, where long sum of small quantities are lost to rouding



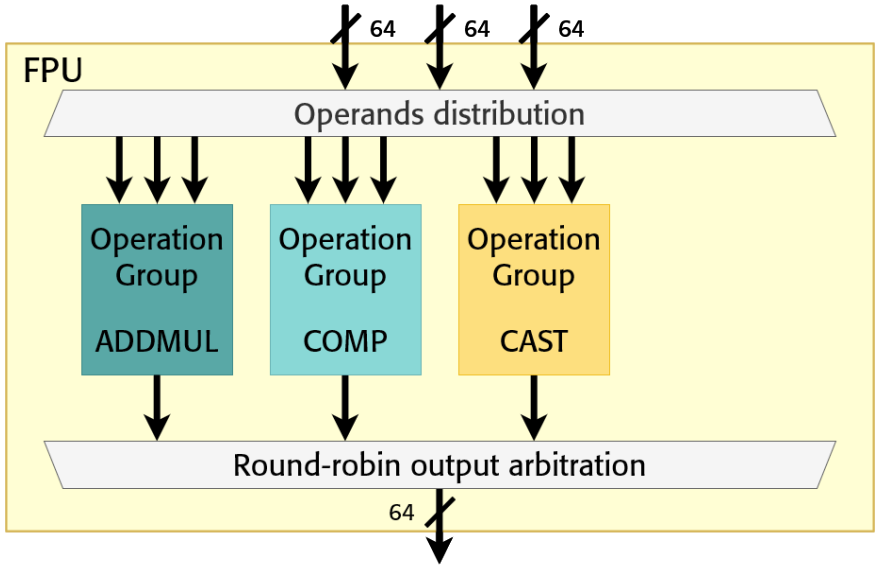
Expanding Sum of Dot Product Unit

- *w = source format bitwidth*
- *2w = destination format bitwidth*
- ExSdotp = $a_w*b_w + c_w*d_w + e_{2w}$
- ExVsum = $a_w*1 + c_w*1 + e_{2w}$
- Vsum = $a_{2w} + c_{2w} + e_{2w}$

- ExVsum/Vsum to reduce and accumulate the results packed in a register after SIMD ExSdotp executions
- Support for non-expanding three-term sum added by **bypassing** the multiplications
- All the necessary logic is already present as the targeted ExSdotp operations were expanding

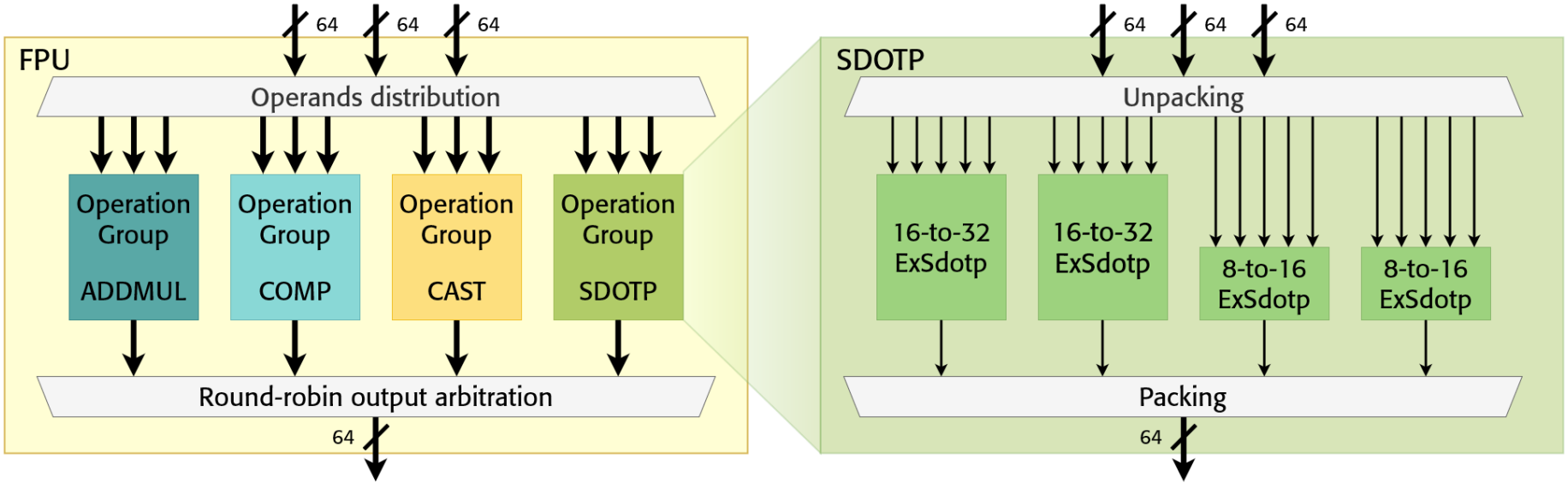


Enhancing CVFPU with SIMD ExSdotp



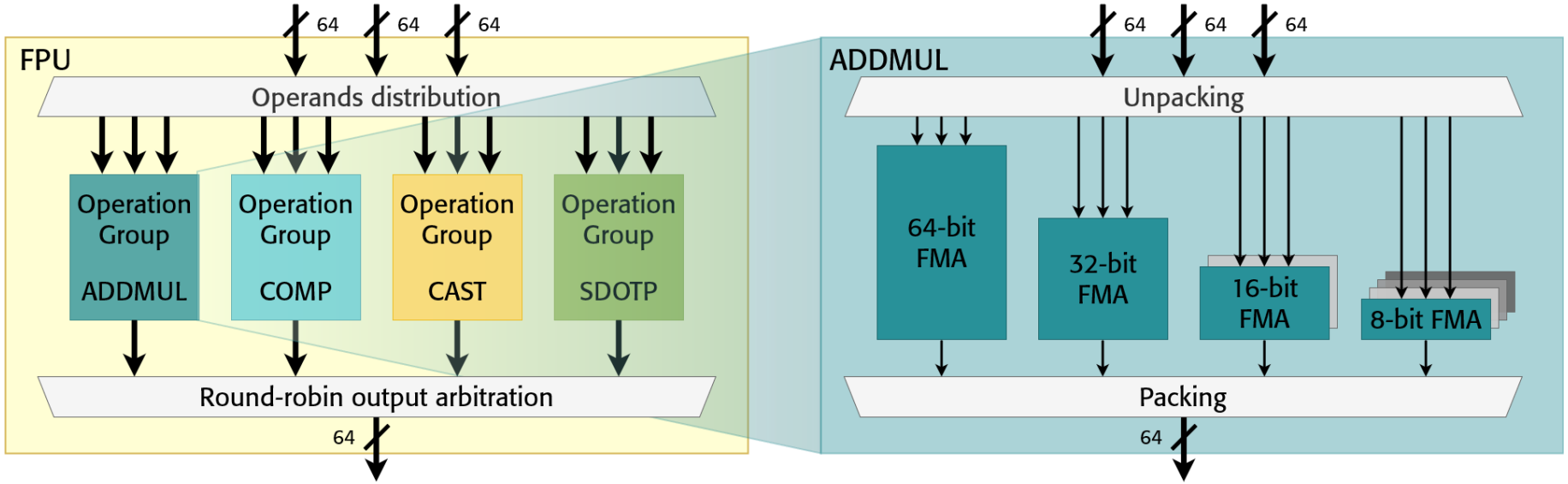
- **CVFPU** is a highly-parameterized open-source **modular** energy-efficient multi-format FPU

Enhancing CVFPU with SIMD ExSdotp



- **CVFPU** is a highly-parameterized open-source **modular** energy-efficient multi-format FPU
- **SIMD ExSdotp** unit integrated into CVFPU as a new **operation group** block
- Parametrized pipeline levels
- SIMD SDOTP: **two** 16-to-32-bit units and **two** 8-to-16-bit units → Up to **two** 16-to-32-bit ExSdotp and **four** 8-to-16-bit **ExSdotp per cycle**

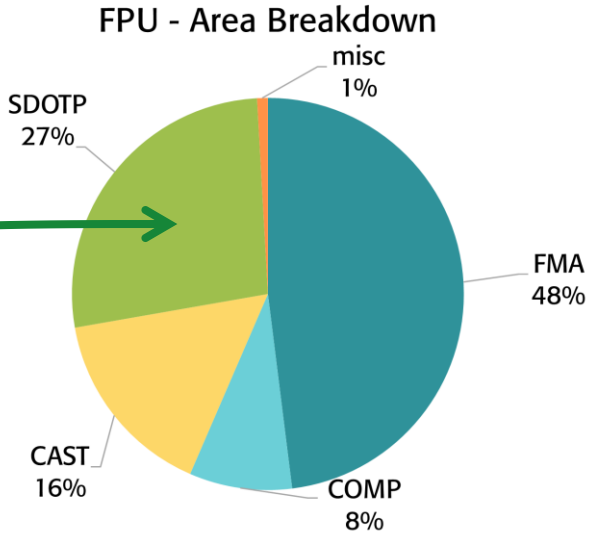
Enhancing CVFPU with SIMD ExSdotp



- **CVFPU** is a highly-parameterized open-source **modular** energy-efficient multi-format FPU
- **SIMD FMA**
- As proposed in *https://iis-git.ee.ethz.ch/smach/smallFloat-spec*

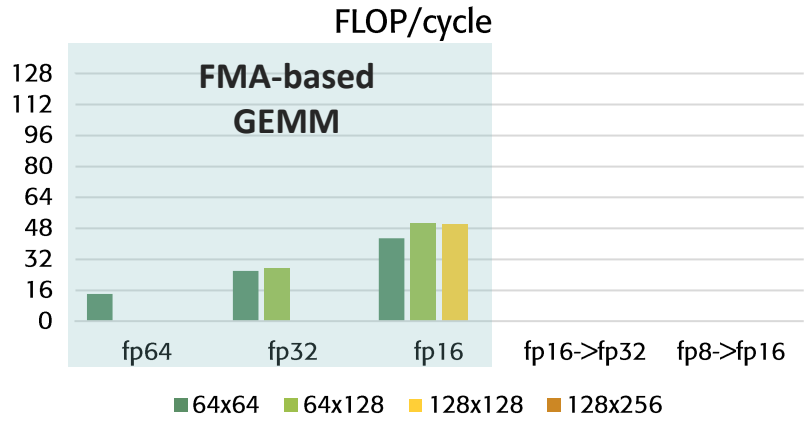
ExSdotp & CVFPU: Area and Timing

- Implemented in GlobalFoundries 12nm FinFET technology
- Max Frequency → **1.24GHz** (typ 0.8V, 25 °C)

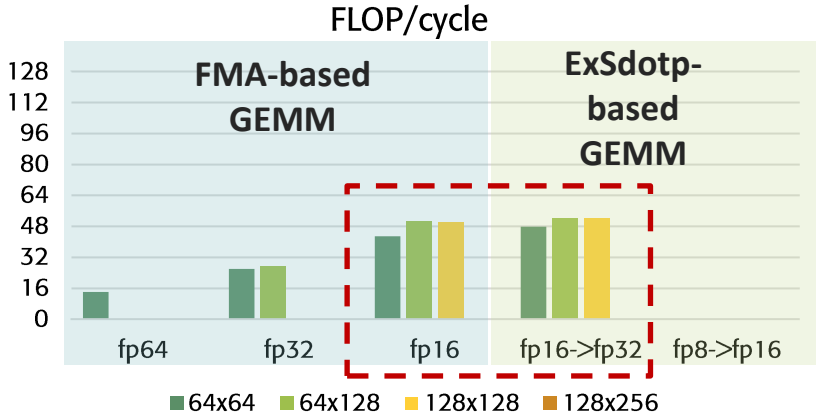


- The SIMD SDOTP unit occupies **44.5 kGE,** amounting to **27%** of the enhanced FPU area (overall FPU area = 165kGE).
- Full extension introduced **less than 15% area overhead at a cluster level**

MiniFloat-NN Cluster: Performance and Efficiency

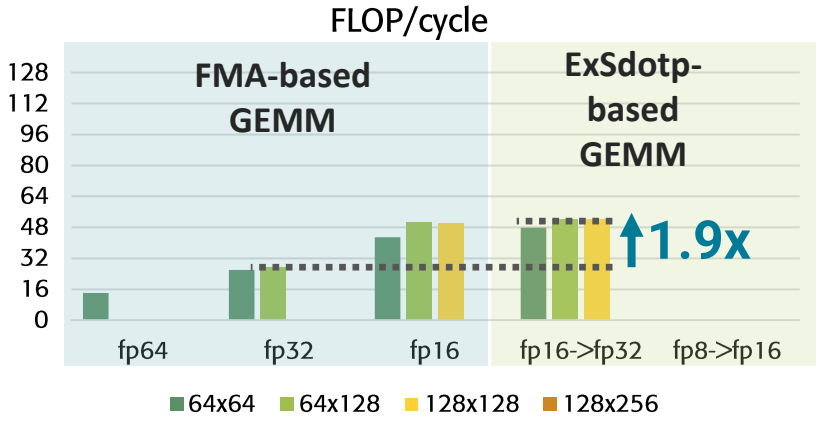


MiniFloat-NN Cluster: Performance and Efficiency

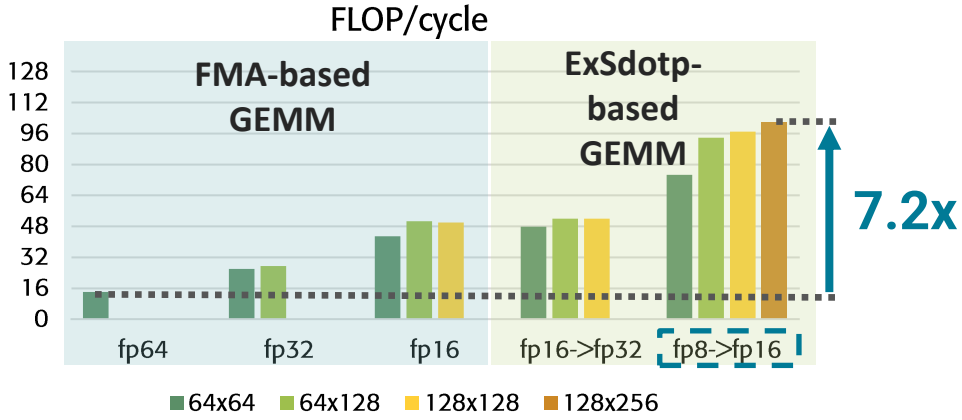


Same performance at a higher accumulation precision

MiniFloat-NN Cluster: Performance and Efficiency



MiniFloat-NN Cluster: Performance and Efficiency



With the required higher precision accumulation

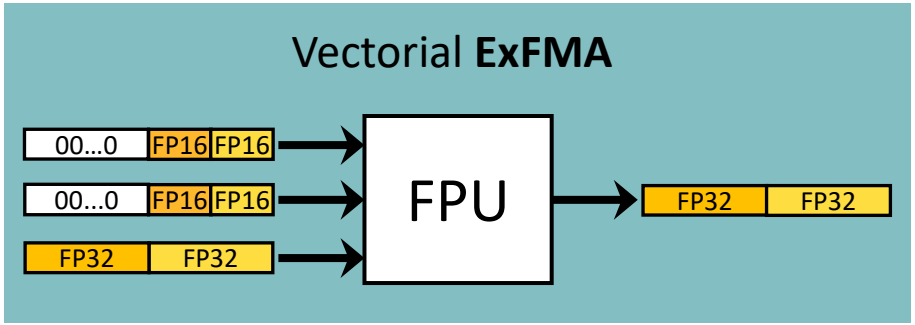
Up to **7.2x performance** and **more than 7.5x energy efficiency** improvement with respect to FP64 computation (<15% area overhead on the entire cluster) + **reduced memory footprint**

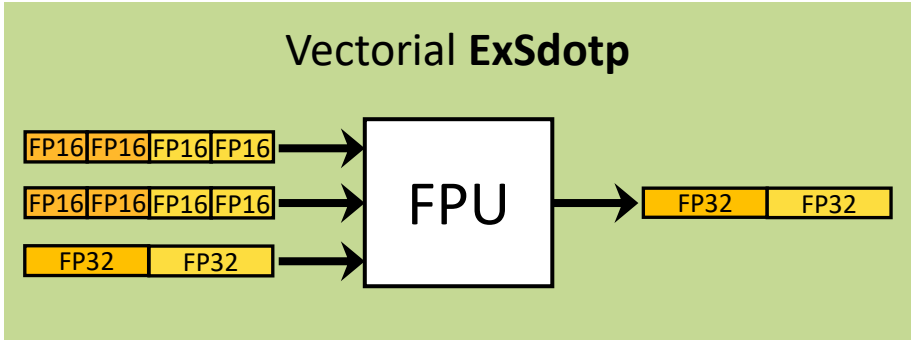
Late-Breaking Results: ExSdotp in the Vector Extension

```
vfwmacc.vv v0, v1, v2
v0_2w[i] += v1_w[i] * v2_w[i]
```

- **Mixed-precision support**
- **Lower memory footprint**
- Still Expanding FMA **underutilize the FPU bandwidth**
- We can provide the FPU with more data and compute more every cycle

```
vfwdotp.vv v0,v1,v2
v0_2w[i] += v1_w[2i]   * v2_w[2i] +
            v1_w[2i+1] * v2_w[2i+1]
```

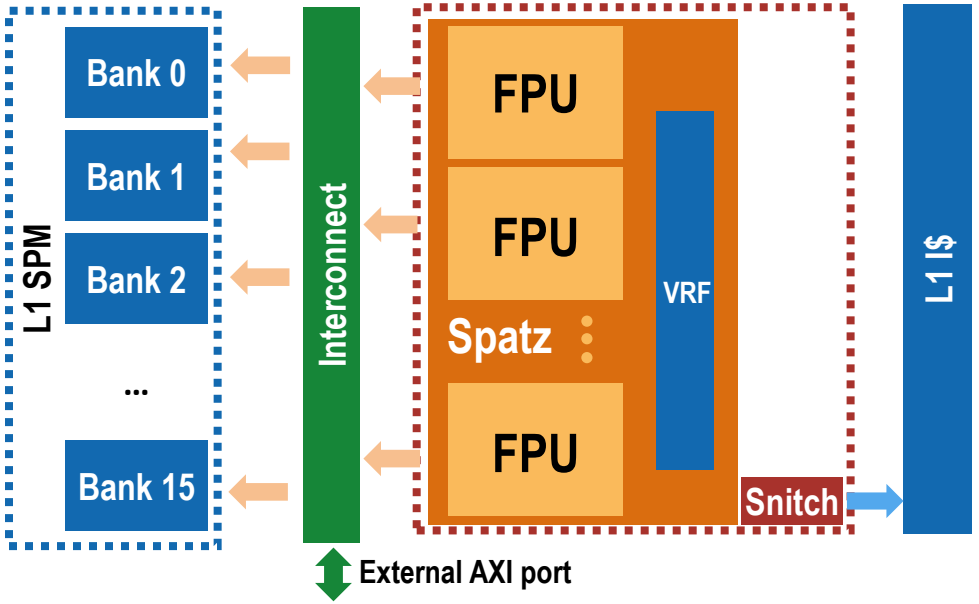




Spatz: a Compact RISC-V Vector Processing Element

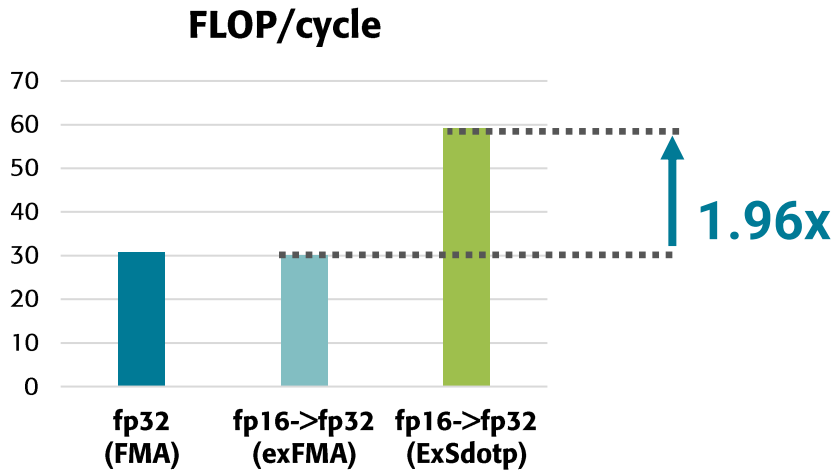
Spatz – a compact vector processor conceived as an alternative building block for the Manticore architecture

- Small latch-based Vector Register File
- **~95% FPU utilization, SIMD FPUs** (FP64-FP8) with ExSdotp support



Cavalcante, M., Wüthrich, D., Perotti, M., Riedel, S., & Benini, L.: Spatz: A Compact Vector Processing Unit for High-Performance and Energy-Efficient Shared-L1 Clusters.", ICCAD 2022 (https://arxiv.org/abs/2207.07970)

MiniFloat-NN Spatz: Performance and Efficiency



Around 2x higher performance with ExSdotp at 1.5x energy efficiency

References

Bertaccini, L., Paulin, G., Fischer, T., Mach, S., Benini, L.:
MiniFloat-NN and ExSdotp: An ISA Extension and a Modular Open Hardware Unit for Low-Precision Training on RISC-V Cores.

ARITH22
(https://arxiv.org/abs/2207.03192)

`github.com/openhwgroup/cvfpu/tree/feature/expanding_dotp`


RISC-V
SUMMIT
@pulp_platform
pulp-platform.org
youtube.com/pulp_platform