A RISC-V ISA Extension for Chaining in Scalar Processors

Luca Colagrande[1], Jayanth Jonnalagadda[2], Luca Benini[1,3]

[1]Integrated Systems Laboratory (IIS) ETH Zürich; [2]D-ITET, ETH Zürich; [3]DEI, University of Bologna

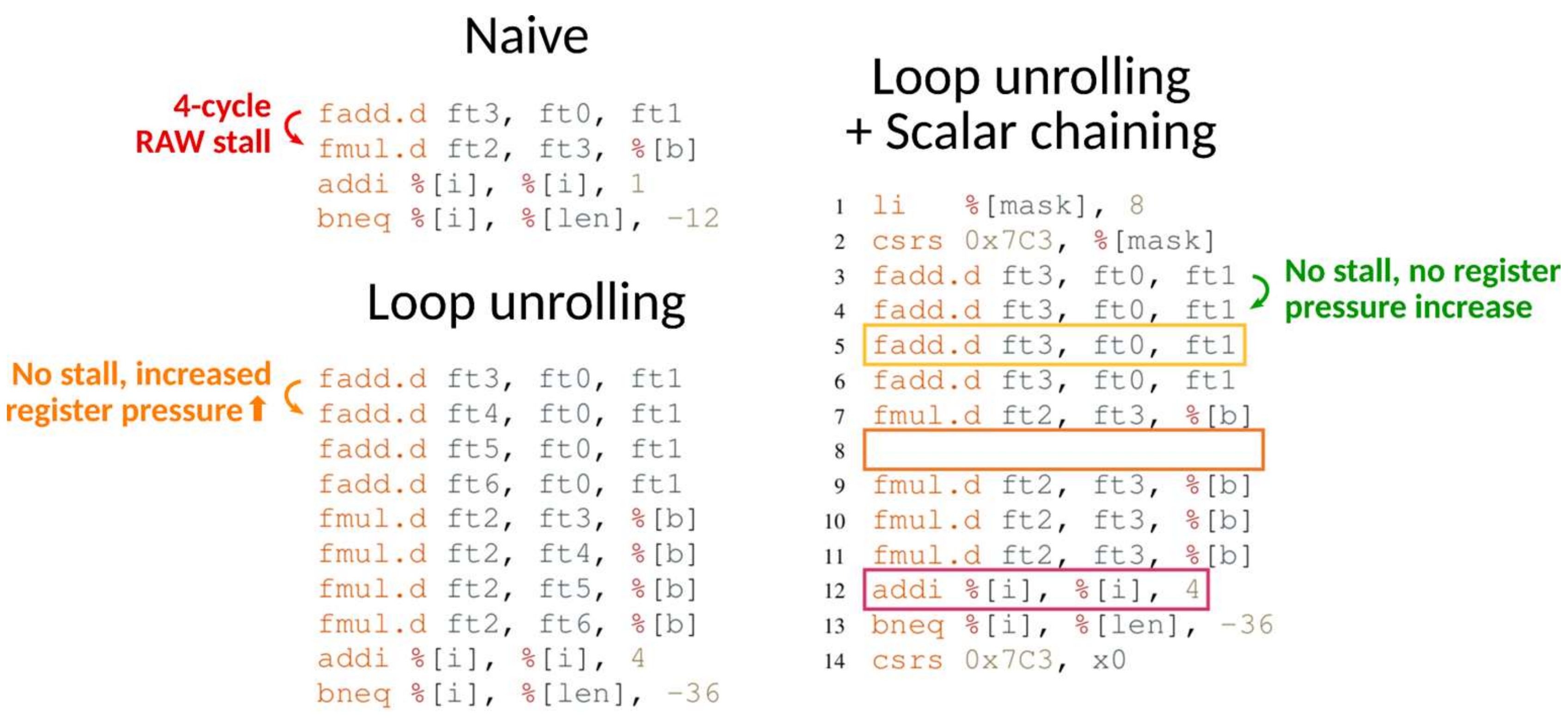
1 Introduction

Modern **general-purpose accelerators** integrate a large number of programmable area- and energy-efficient processing elements (PEs), to deliver high performance while meeting stringent power delivery and thermal dissipation constraints. In this context, PEs are often implemented by **scalar in-order cores**, which are highly sensitive to **pipeline stalls**. Traditional software techniques, such as loop unrolling, mitigate the issue at the cost of increased register pressure, limiting flexibility. We propose **scalar chaining**, a novel hardware-software solution, to address this issue without incurring the drawbacks of traditional software-only techniques. We demonstrate our solution on register-limited stencil codes, achieving **>93% FPU utilizations** and a **4% speedup** and **10% higher energy efficiency**, on average, over highly-optimized baselines.

2 Implementation

We implement **dataflow** (or **FIFO**) **semantics** in the scalar in-order Snitch[1] core, to **chain** functional units (FUs) through the register file (RF) and ensure that values from the producer FU are not overwritten until they are used by the consumer FU.

The producer FU's pipeline registers form a **logical FIFO**, which can be effectively used to store intermediate results from **loop unrolling**, without adding **pressure** on the architectural RF.



Naive

```
fadd.d ft3, ft0, ft1
fmul.d ft2, ft3, %[b]
addi %[i], %[i], 1
bneq %[i], %[len], -12
```

Loop unrolling

No stall, increased register pressure ⬆

```
fadd.d ft3, ft0, ft1
fadd.d ft4, ft0, ft1
fadd.d ft5, ft0, ft1
fadd.d ft6, ft0, ft1
fmul.d ft2, ft3, %[b]
fmul.d ft2, ft4, %[b]
fmul.d ft2, ft5, %[b]
fmul.d ft2, ft6, %[b]
addi %[i], %[i], 4
bneq %[i], %[len], -36
```

Loop unrolling + Scalar chaining

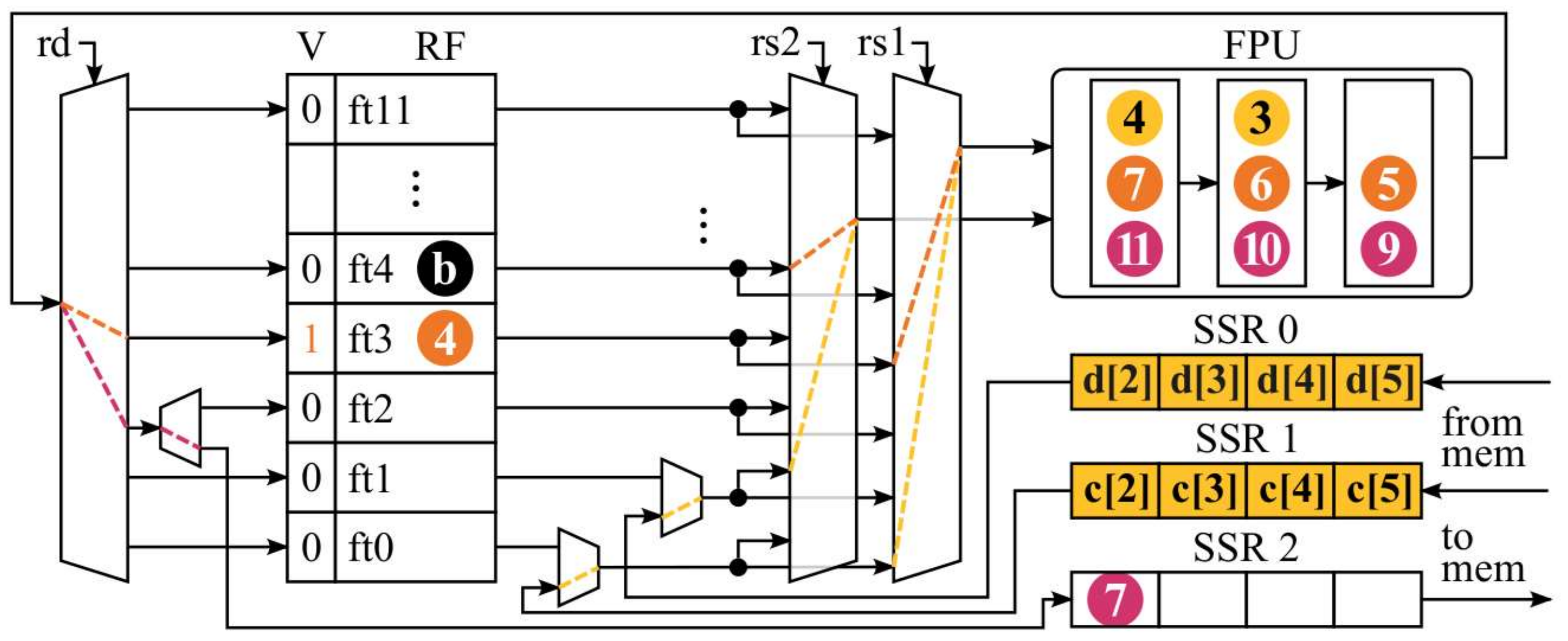
No stall, no register pressure increase

```
li   %[mask], 8
csrs 0x7C3, %[mask]
fadd.d ft3, ft0, ft1
fadd.d ft3, ft0, ft1
fadd.d ft3, ft0, ft1
fadd.d ft3, ft0, ft1
fmul.d ft2, ft3, %[b]

fmul.d ft2, ft3, %[b]
fmul.d ft2, ft3, %[b]
fmul.d ft2, ft3, %[b]
addi %[i], %[i], 4
bneq %[i], %[len], -36
csrs 0x7C3, x0
```

In this example, the `fadd` and `fmul` instructions are chained through `ft3` (the FPU is both the consumer and producer FU). Stream semantics[2] are assigned to `ft0`, `ft1` and `ft2`.

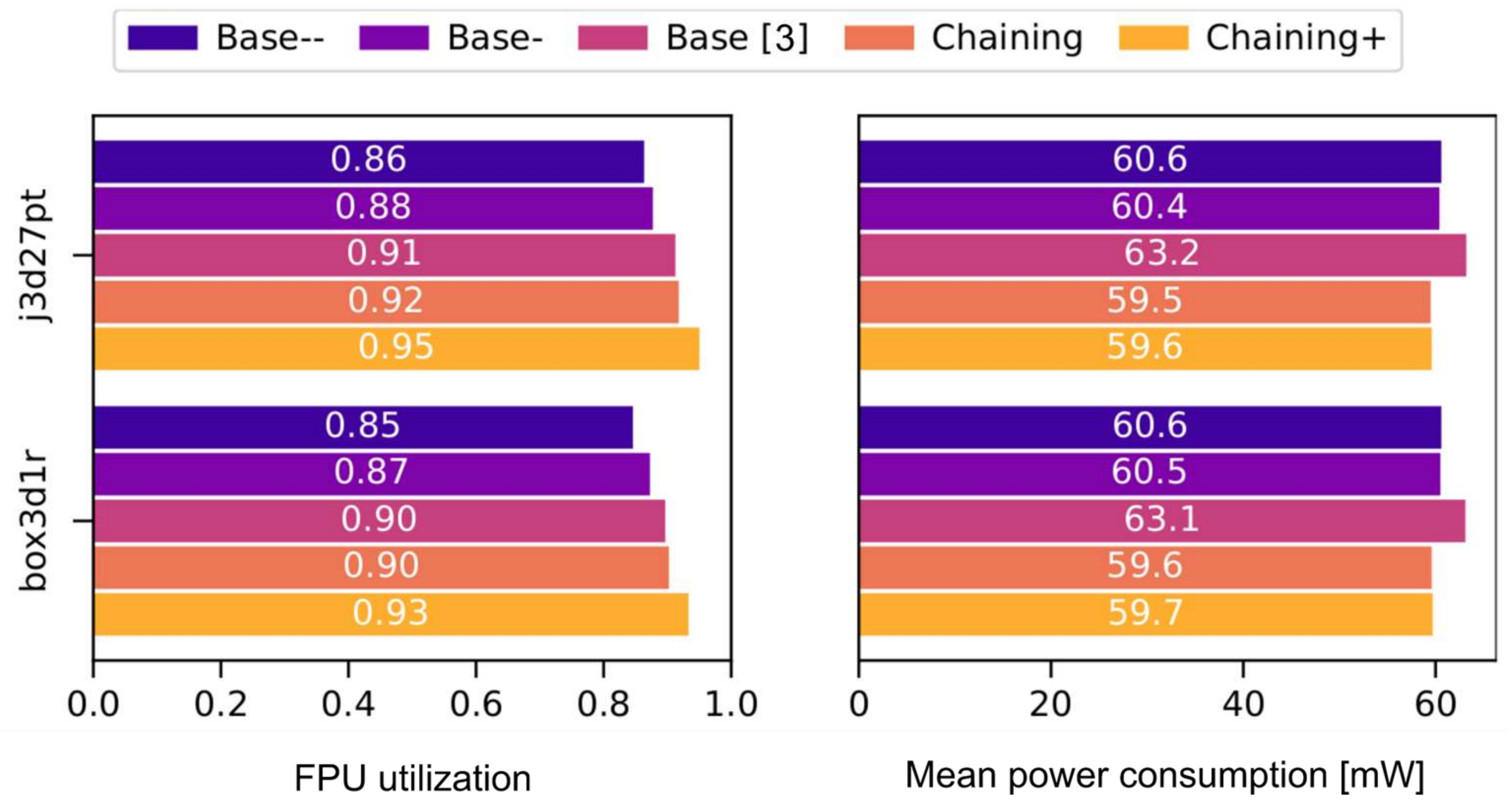
The RF is augmented with a valid bit per register to implement head-of-line blocking at the consumer's side of the logical FIFO.



3 Results and Discussion

On a Snitch cluster implemented in GlobalFoundries' 12LP+ FinFET technology using Fusion Compiler 2023.12, with a target clock frequency of 1 GHz, our extensions introduce **negligible area and timing overheads**, in the scale of synthesis process variability margins.

We evaluate our implementation on two register-limited stencil codes[3], `box3d1r` and `j3d27pt`. By applying chaining, we can free enough registers to fully store the stencil coefficients in the RF, achieving a **4% speedup** and **10% higher energy efficiency**, on average, over the highly optimized baselines in [3], and **>93% FPU utilizations**.



4 Conclusion

We presented a novel hardware and software solution to hide FU latencies in scalar in-order processors, without incurring increased register pressure, as with traditional software-only techniques. With a negligible area and timing cost, our solution is lightweight and suited for integration into highly area- and energy-efficient cores.

References

1. F. Zaruba et al., "Snitch: A tiny pseudo dual-issue processor for area and energy efficient execution of floating-point intensive workloads," IEEE Transactions on Computers, vol. 70, no. 11, pp. 1845–1860, 2021.
2. F. Schuiki et al., "Stream semantic registers: A lightweight risc-v isa extension achieving full compute utilization in single-issue cores," IEEE Trans. Comput., vol. 70, pp. 212–227, 2021.
3. P. Scheffler et al., "Saris: Accelerating stencil computations on energy-efficient risc-v compute clusters with indirect stream registers," in DAC'24: Proceedings of the 61st ACM/IEEE Design Automation Conference.