POLITECNICO
MILANO 1863

SALSA
A Domain Specific Architecture for Sequence Alignment

27th IEEE Reconfigurable Architecture Workshop
May 18th, 2020
New Orleans, Lousiana USA

Lorenzo Di Tucci lorenzo.ditucci@polimi.it - lditucci@csail.mit.edu
Riyadh Baghdadi baghdadi@csail.mit.edu
Saman Amarasinghe saman@csail.mit.edu
Marco D. Santambrogio marco.santambrogio@polimi.it

Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB) & Computer Science and Artificial Intelligence Laaboratory (CSAIL)
Politecnico di Milano - MIT

POLITECNICO MILANO 1863 NECST laboratory

Context

- The explosion of genomic data is fostering research in fields such as personalized medicine and agritech[1]

- Moore’s Law and Dennard Scaling are ending, and there is the necessity to provide more performant, power-efficient and easy to use architectures [2]

- GPUs and FPGAs delivers major performance improvements on specific applications, however, GPUs present notable power consumption, while FPGA lacks programmability

[1] G. S. Ginsburg and J. J. McCarthy, “Personalized medicine: revolu- tionizing drug discovery and patient care,” *TRENDS in Biotechnology*, vol. 19, no. 12, pp. 491–496, 2001.
[2] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach*. Elsevier, 2017

Contributions

The design and evaluation of SALSA, a Domain Specific Architecture (DSA) for sequence alignment that is:

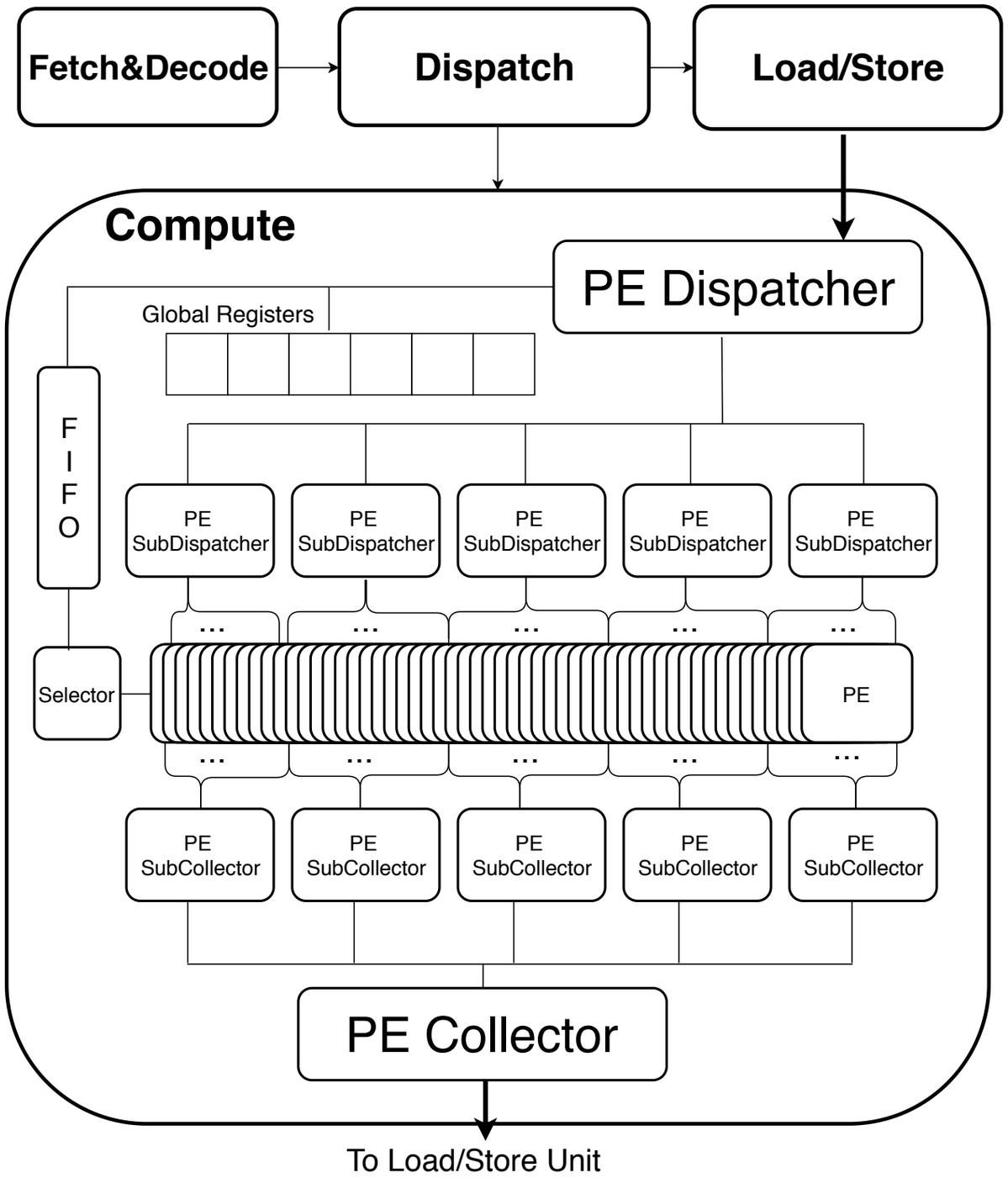
- **Highly Programmable:** thanks to RISC-V ISA[1], with the possibility to extend the Instruction Set by introducing custom instructions;

- **Customizable:** it is possible to tune different parameters in the source code to generate architecture with different features like the number of processing elements, or the size of the memory port;

- **Extensible:** each Processing Element is composed of a general purpose ALU programmable via software, and more specific ALUs. The user can design and integrate custom ALUs inside SALSA, exploiting the overall architecture

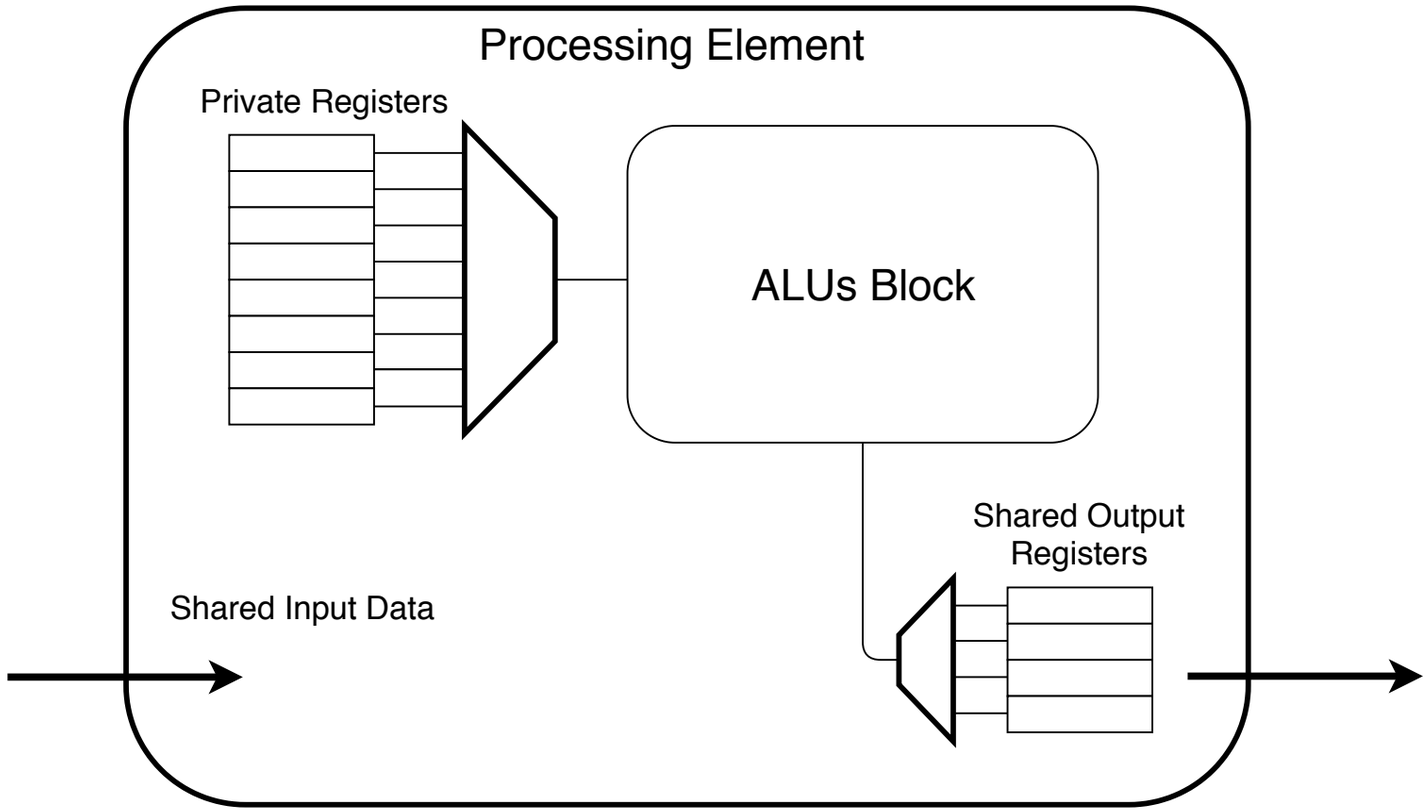
[1] J. Bachrach, H. Vo, B. Richards, Y. Lee, A. Waterman, R. Aviz˘ienis, J. Wawrzynek, and K. Asanovic´, “Chisel: constructing hardware in a scala embedded language,” in *DAC Design Automation Conference 2012*. IEEE, 2012, pp. 1212–1221.

SALSA
Top Level Architecture



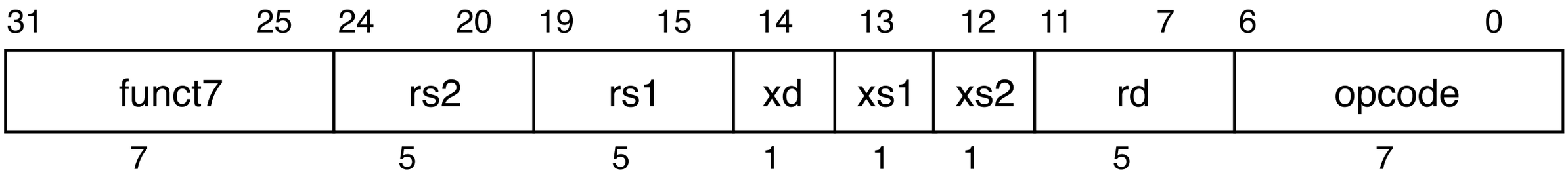
- Optimized datapath for handling reduced-bitwidth variables
- Compute step composed of a systolic array of processing elements to handle wavefront computations
- Detached Compute-Load/Store units with a stalling mechanism to handle memory transactions
- Hierarchical PE Dispatchers to avoid routing congestions
- Possibility to send data to all Pes as well as to single Pes thanks to dispatchers and global registers

Processing Element



- PEs communications thanks to shared input/output registers
- The ALUs block contains General Purpose ALUs and Specific Purpose ALUs to perform multiple operations within the single clock cycle
- Possibility to design and integrated Specific Purpose ALU to benefit from the overall infrastructure

RISC-V ISA Extension



Registers for the RISC-V ISA extension by ROCC has been re-purposed to introduce new instructions such as loading multiple variables from memory with a single instruction, load multiple reduced-bitwidth variables in a 64 bits one or handling the specific purpose ALUs available or designed by the user.

Experimental Settings

- SALSA has been designed using **Chisel HD**L, integrated in Rocket using the **ROCC Interface**[1], with direct access to Rocket Cache (64bits wide) and implemented on a Xilinx Virtex Ultrascale+ FPGA.

- SALSA architectural parameters: 160 Processing Elements, 16 32-bits *global* registers, 5 *sub-dispatchers* and 5 *sub-collectors*. Each PE features 20 32-bits *private* registers, 6 32-bits *shared output* registers

- The design and implementation of SALSA have been done with **Firesim 1.5**[2], testing on a single AWS EC2 F1 instance at a **target frequency of 200 MHz**

- Performance test performed on 4 applications: **Smith-Waterman** with constant and affine-gap penalty, **Needleman Wunsch** and **MaxScore**

- The application has been tested on **Rocket (1GHz and 3GHz)**, **SALSA (200Mhz)** and on an **Intel Xeon E3 (3.7GHz)** running **SeqAn**[3] as a software baseline.

- Performance benchmarked by means of Giga Cell Update per Second (**GCUPS**)

[1] C. Schmidt, “RISC-V Rocket-Chip Tutorial,” https://riscv.org/wp- content/uploads/2015/01/riscv-rocket-chip-tutorial-bootcamp- jan2015.pdf, 2015,
[2] S. Karandikar, H. Mao, D. Kim, D. Biancolin, A. Amid, D. Lee, N. Pemberton, E. Amaro, C. Schmidt, A. Chopra *et al.*, “Firesim: Fpga-accelerated cycle-exact scale-out system simulation in the public cloud,” in *Proceedings of the 45th Annual International Symposium on Computer Architecture*. IEEE Press, 2018, pp. 29–42.
[3] A. Do¨ring, D. Weese, T. Rausch, and K. Reinert, “Seqan an efficient, generic c++ library for sequence analysis,” *BMC bioinformatics*, vol. 9, no. 1, p. 11, 2008.

Results

Algorithm	Dataset	SALSA exec. time @ 200Mhz [μs]	Salsa Performance Improvement w.r.t. Rocket @ 1 GHz	w.r.t. Rocket @ 3 GHz	w.r.t. CPU @ 3.7 GHz
MaxScore	32x64	0.8	**101.66x**	**33.89x**	**47.64x**
MaxScore	64x128	1.045	**350.07x**	**116.69x**	**96.74x**
SmithWaterman	32x64	16.84	**4.99x**	**1.66x**	**2.38x**
SmithWaterman	64x128	37.01	**9.86x**	**3.28x**	**3.31x**
NeedlemanWunsch	32x64	16.71	**4.85x**	**1.62x**	**1.99x**
NeedlemanWunsch	64x128	37.00	**9.56x**	**3.19x**	**2.76x**
SmithWatermanAffine	32x64	17.8	**8.62x**	**2.87x**	**2.55x**
SmithWatermanAffine	64x128	39.48	**17.45x**	**5.82x**	**4.04x**

- Execution Times includes data movement

- SALSA is faster than Rocket & SeqAn on the Intel Xeon in all the applications benchmarked, with performance improvement up to more than **300 times**

- When exploiting 40 threads on the Intel Xeon E3, SeqAn obtains 2GCUPS, while SALSA's best performance (MaxScore example) obtain 8GCUPS

Results

Algorithm	Dataset	Power Efficiency [$GCUPS/W$] SALSA	CPU	Improvement
MaxScore	32x64	231.28	0.59	**389.8x**
MaxScore	64x128	712.66	0.9	**791.5x**
SmithWaterman	32x64	11.06	0.56	**19.5x**
SmithWaterman	64x128	20.12	0.74	**27.13x**
NeedlemanWunsch	32x64	11.14	0.68	**16.34x**
NeedlemanWunsch	64x128	20.13	0.89	**22.61x**
SmithWatermanAffine	32x64	10.45	0.5	**20.93x**
SmithWatermanAffine	64x128	18.86	0.57	**33.09x**

Performance Efficiency comparison by means of GCUPS/W

SALSA is always more power efficient than the Intel processor, with improvements up to **790 times** (MaxScore example)

Conclusions

This paper presents **SALSA** a **DSA for Sequence Alignment**

- Good tradeoff of performance, power consumption and programmability
- **Chisel HDL** and **RISC-V ISA** make it highly **programmable**, **parametrizable**, **customizable** and **extensible**
- Possibility to **modify the ALU to specialize the computation** exploiting the overall architecture
- Performance comparison showed **790x power efficiency** improvement and **60x performance improvement**
- As SALSA's architecture is based on Systolic Arrays (SAs), future works will involve the exploration of its usage in other contexts such as Convolutional Neural Networks and the introduction of a bi-dimensional SA.

For questions regarding this work, email:
Lorenzo Di Tucci: lorenzo.ditucci@polimi.it – lditucci@csail-mit.edu