

EPI Forum

Barcelona, 9–10.10.2024.





The EPI Accelerator (EPAC): A collection of RISC-V based accelerators

Filippo Mantovani, Barcelona Supercomputing Center (BSC)

What is RISC-V?



What is RISC-V?

Core Instruction Formats

31	27	26	25	24	20	19	15	14	12	11	7	6	0		
funct7				rs2		rs1		funct3			rd		opcode		R-type
imm[11:0]						rs1		funct3			rd		opcode		I-type
imm[11:5]				rs2		rs1		funct3			imm[4:0]		opcode		S-type
imm[12:10:5]				rs2		rs1		funct3			imm[4:1:11]		opcode		B-type
				imm[31:12]							rd		opcode		U-type
				imm[20:10:1:11:19:12]							rd		opcode		J-type

RV32I Base Integer Instructions

Inst.	Name	FMT	Opcode	funct3	funct7	Description (C)	Note
add	ADD	R	0110011	0x0	0x00	rd = rs1 + rs2	
sub	SUB	R	0110011	0x0	0x20	rd = rs1 - rs2	
xor	XOR	R	0110011	0x4	0x00	rd = rs1 ^ rs2	
or	OR	R	0110011	0x6	0x00	rd = rs1 rs2	
and	AND	R	0110011	0x7	0x00	rd = rs1 & rs2	
sll	Shift Left Logical	R	0110011	0x1	0x00	rd = rs1 << rs2	
srl	Shift Right Logical	R	0110011	0x5	0x00	rd = rs1 >> rs2	
sra	Shift Right Arith*	R	0110011	0x5	0x20	rd = rs1 >> rs2	msb-extends
slt	Set Less Than	R	0110011	0x2	0x00	rd = (rs1 < rs2)?1:0	
sltu	Set Less Than (U)	R	0110011	0x3	0x00	rd = (rs1 < rs2)?1:0	zero-extends
addi	ADD Immediate	I	0010011	0x0		rd = rs1 + imm	
xori	XOR Immediate	I	0010011	0x4		rd = rs1 ^ imm	
ori	OR Immediate	I	0010011	0x6		rd = rs1 imm	
andi	AND Immediate	I	0010011	0x7		rd = rs1 & imm	
slli	Shift Left Logical Imm	I	0010011	0x1	imm[5:11]=0x00	rd = rs1 << imm[0:4]	
srl	Shift Right Logical Imm	I	0010011	0x5	imm[5:11]=0x00	rd = rs1 >> imm[0:4]	
srai	Shift Right Arith Imm	I	0010011	0x5	imm[5:11]=0x20	rd = rs1 >> imm[0:4]	msb-extends
slti	Set Less Than Imm	I	0010011	0x2		rd = (rs1 < imm)?1:0	
sltiu	Set Less Than Imm (U)	I	0010011	0x3		rd = (rs1 < imm)?1:0	zero-extends
lb	Load Byte	I	0000011	0x0		rd = M[rs1+imm][0:7]	
lh	Load Half	I	0000011	0x1		rd = M[rs1+imm][0:15]	
lw	Load Word	I	0000011	0x2		rd = M[rs1+imm][0:31]	
lbu	Load Byte (U)	I	0000011	0x4		rd = M[rs1+imm][0:7]	zero-extends
lhu	Load Half (U)	I	0000011	0x5		rd = M[rs1+imm][0:15]	zero-extends
sb	Store Byte	S	0100011	0x0		M[rs1+imm][0:7] = rs2[0:7]	
sh	Store Half	S	0100011	0x1		M[rs1+imm][0:15] = rs2[0:15]	
sw	Store Word	S	0100011	0x2		M[rs1+imm][0:31] = rs2[0:31]	
beq	Branch ==	B	1100011	0x0		if(rs1 == rs2) PC += imm	
bne	Branch !=	B	1100011	0x1		if(rs1 != rs2) PC += imm	
blt	Branch <	B	1100011	0x4		if(rs1 < rs2) PC += imm	
bge	Branch ≥	B	1100011	0x5		if(rs1 ≥ rs2) PC += imm	
bltu	Branch < (U)	B	1100011	0x6		if(rs1 < rs2) PC += imm	zero-extends
bgeu	Branch ≥ (U)	B	1100011	0x7		if(rs1 ≥ rs2) PC += imm	zero-extends
jal	Jump And Link	J	1101111			rd = PC+4; PC += imm	
jalr	Jump And Link Reg	I	1101111	0x0		rd = PC+4; PC = rs1 + imm	
lui	Load Upper Imm	U	0110111			rd = imm << 12	
auipc	Add Upper Imm to PC	U	0010111			rd = PC + (imm << 12)	
ecall	Environment Call	I	1110011	0x0	imm=0x0	Transfer control to OS	
ebreak	Environment Break	I	1110011	0x0	imm=0x1	Transfer control to debugger	

- Instruction Set Architecture (ISA)
 - Simple and modular
- Research project started at Berkeley in 2010
 - ratified ISA in 2014
- RISC-V (pronounced “risc five”, as it is the fifth generation of RISC ISA at Berkeley)

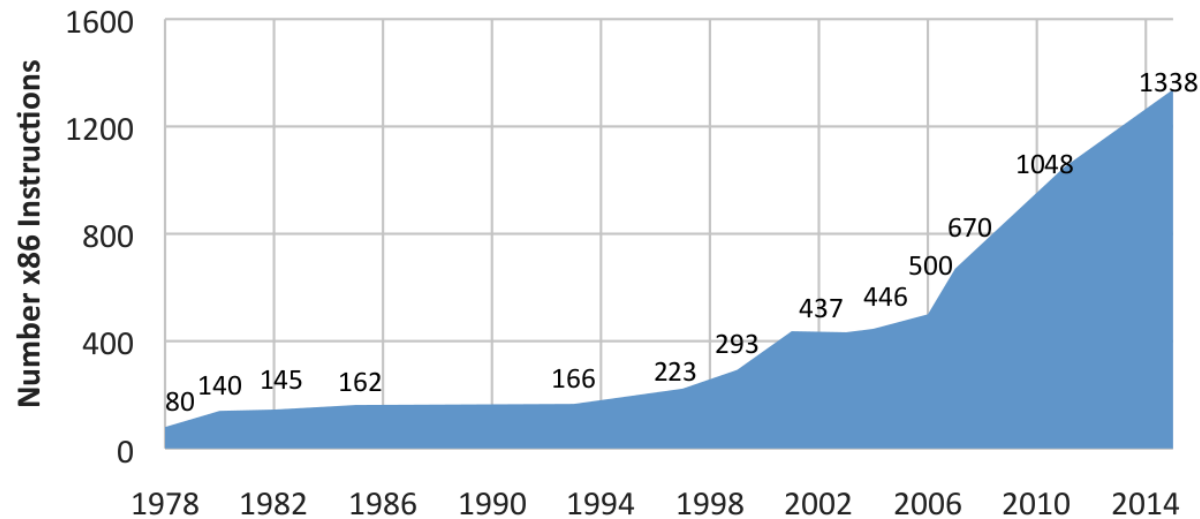


Waterman, Andrew., Patterson, David A.. **The RISC-V Reader: An Open Architecture Atlas**. United States: Strawberry Canyon LLC, 2017.

Patterson, David A., Hennessy, John L.. **Computer Organization and Design RISC-V Edition: The Hardware Software Interface**. Netherlands: Elsevier Science, 2017.

Incremental vs modular ISA

- Intel x86 is an incremental ISA. Each new release:
 - Maintain backward compatibility
 - Carry on new instructions (also for marketing reasons)



https://en.wikichip.org/wiki/risc-v/standard_extensions

Name	Description	Version	Status	Instruction Count
RV32I	Base Integer Instruction Set - 32-bit	2.1	Frozen	49
RV32E	Base Integer Instruction Set (embedded) - 32-bit, 16 registers	1.9	Open	Same as RV32I
RV64I	Base Integer Instruction Set - 64-bit	2.0	Frozen	14
RV128I	Base Integer Instruction Set - 128-bit	1.7	Open	14
Extension				
M	Standard Extension for Integer Multiplication and Division	2.0	Frozen	8
A	Standard Extension for Atomic Instructions	2.0	Frozen	11
F	Standard Extension for Single-Precision Floating-Point	2.0	Frozen	25
D	Standard Extension for Double-Precision Floating-Point	2.0	Frozen	25
G	Shorthand for the base and above extensions	n/a	n/a	n/a
Q	Standard Extension for Quad-Precision Floating-Point	2.0	Frozen	27
L	Standard Extension for Decimal Floating-Point	0.0	Open	Undefined Yet
C	Standard Extension for Compressed Instructions	2.0	Frozen	36
B	Standard Extension for Bit Manipulation	0.90	Open	42
J	Standard Extension for Dynamically Translated Languages	0.0	Open	Undefined Yet
T	Standard Extension for Transactional Memory	0.0	Open	Undefined Yet
P	Standard Extension for Packed-SIMD Instructions	0.1	Open	Undefined Yet
V	Standard Extension for Vector Operations	0.7	Open	186
N	Standard Extension for User-Level Interrupts	1.1	Open	3
H	Standard Extension for Hypervisor	1.0	Frozen	2
S	Standard Extension for Supervisor-level Instructions	1.12	Open	7

Another business model



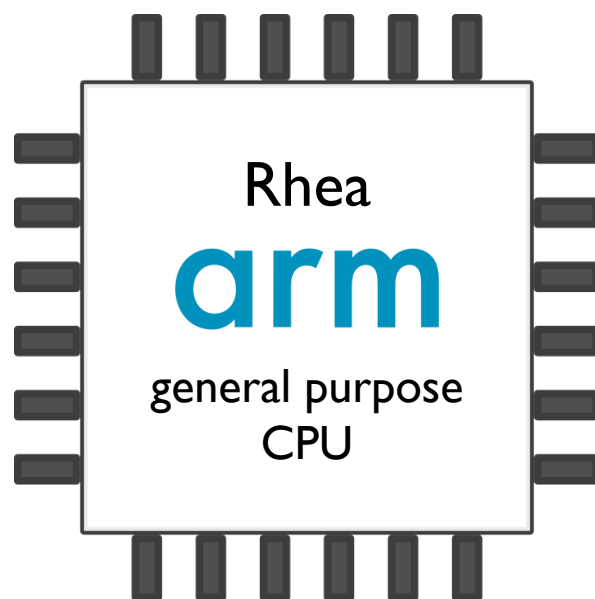
arm



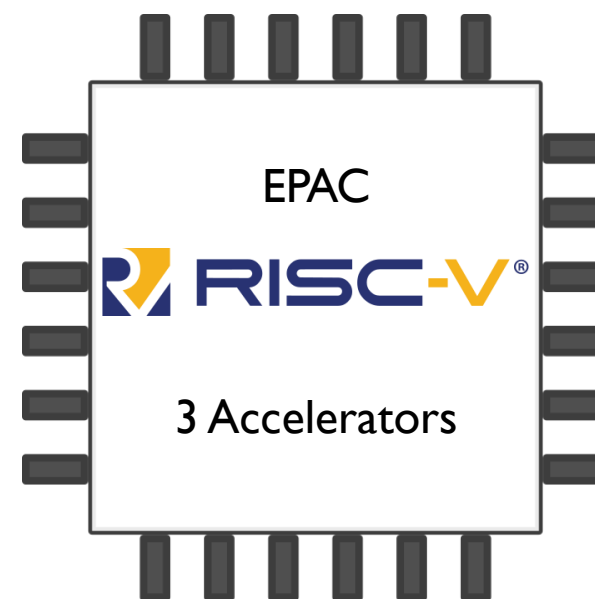
Closed ISA		Closed ISA		Open ISA
Adopting the ISA is not possible (only a few partners can)		Adopting the ISA requires paying a fee		Adopting the ISA is free
No access to the architecture, core IPs, instructions, extensions, etc.		Commercial IPs by one vendor		Commercial IPs by several vendors
		Several implementations		Several implementations
		Almost no access to extension		Full access to extensions
		No open-source IP cores		Possibility of open-source IP cores

EPI Main Objective

- To develop European microprocessor and accelerator technology
- Strengthen competitiveness of EU industry and science



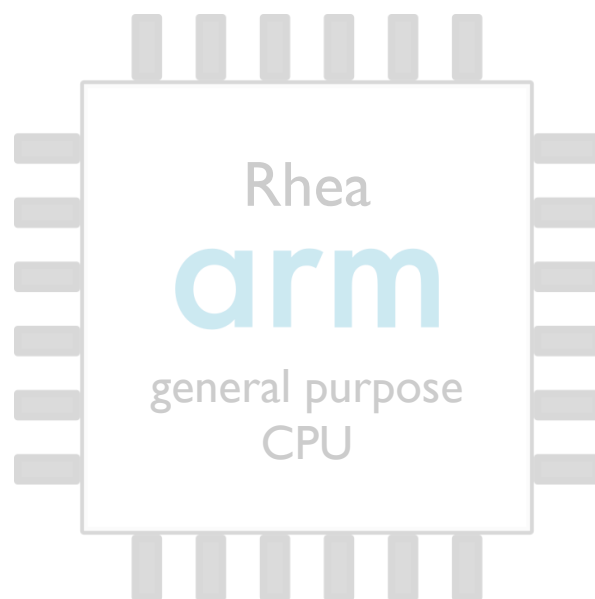
SiPearl, Atos, CEA, UniBo,
E4, UniPi, P&R



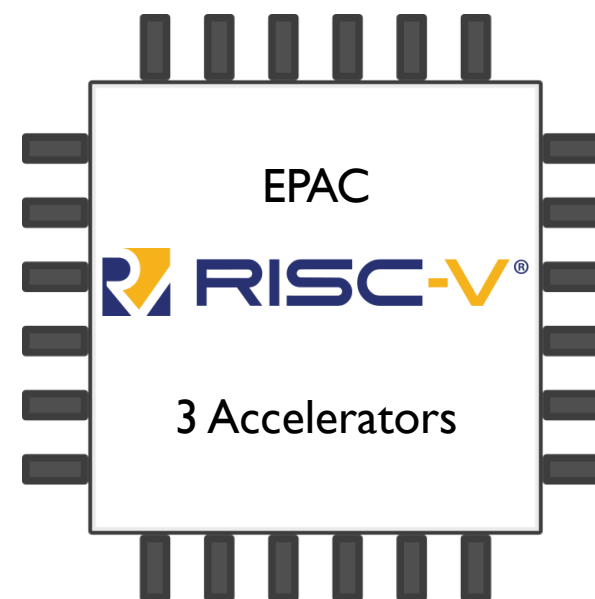
BSC, SemiDynamics, EXTOLL, FORTH,
ETHZ, UniBo, UniZG, Chalmers, CEA, E4

EPI Main Objective

- To develop European microprocessor and accelerator technology
- Strengthen competitiveness of EU industry and science



SiPearl, Atos, CEA, UniBo,
E4, UniPi, P&R



BSC, SemiDynamics, EXTOLL, FORTH,
ETHZ, UniBo, UniZG, Chalmers, CEA, E4

EPAC: EPI Accelerator v1.5

GF22FDX, 27 mm², 0.3 Btr Tape out Mar 2023, Bring up Oct 2023

VEC tile

General purpose RISC-V CPU
Avispado Core (16 kI\$, 32 kD\$)
with dedicated VPU
Up to 256 DP element vector length



L2-HN tile

Distributed L2 cache (256 kB/slice) and
Coherence Home Node



VRP tile

General purpose RISC-V CPU
supporting variable precision
arithmetic up to 256 bit elements



STX tile

RISC-V many-core machine learning
accelerator targeting stencil and
tensor arithmetics.



CHI NoC and SerDes

On-chip high-speed network based
on multiple CHI cross points (XP).

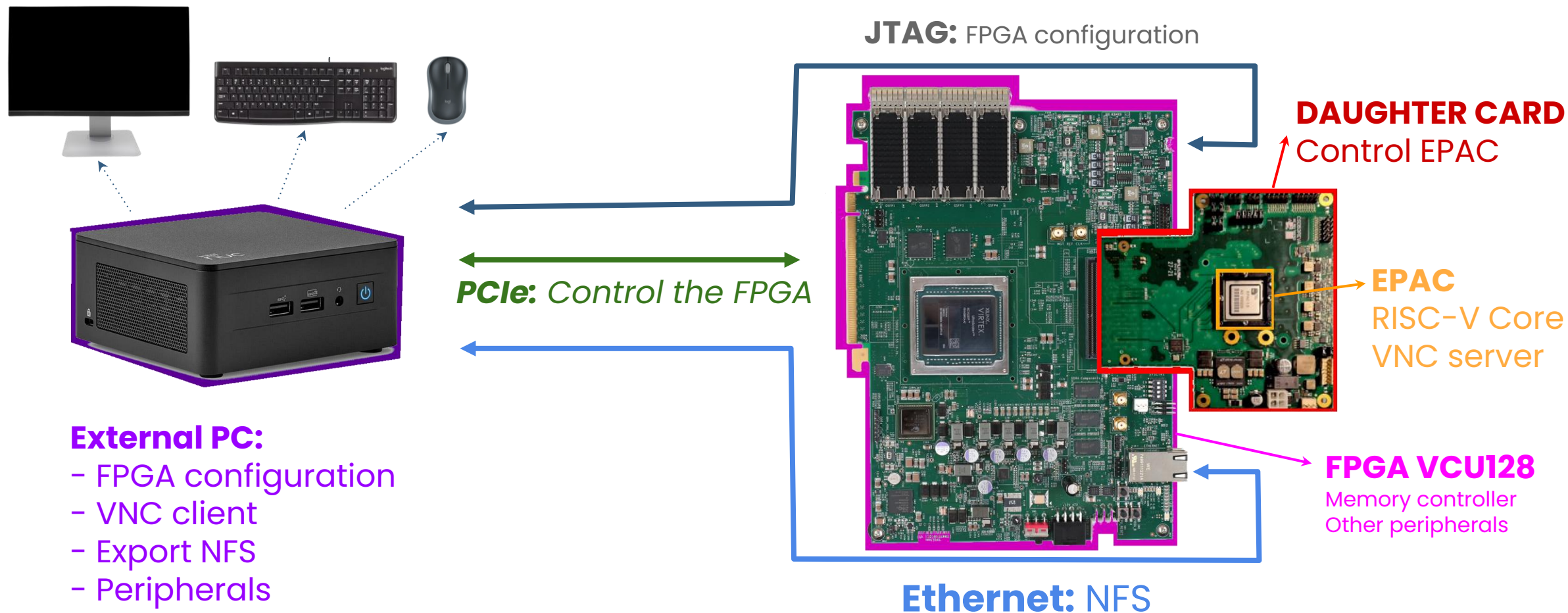
Off-chip link based on SerDes.



Physical design by  Fraunhofer

Prototype board integration by  E4
COMPUTER
ENGINEERING

The DEMO of EPAC is in this room!



EPI Forum

Barcelona, 9–10.10.2024.





The EPAC - VEC:

A RISC-V Vector Accelerator

Filippo Mantovani, Barcelona Supercomputing Center (BSC)

EPAC: EPI Accelerator v1.5

GF22FDX, 27 mm², 0.3 Btr Tape out Mar 2023, Bring up Oct 2023

VEC tile

General purpose RISC-V CPU
Avispado Core (16 kI\$, 32 kD\$)
with dedicated VPU
Up to 256 DP element vector length



L2-HN tile

Distributed L2 cache (256 kB/slice) and
Coherence Home Node



VRP tile

General purpose RISC-V CPU
supporting variable precision
arithmetic up to 256 bit elements



STX tile

RISC-V many-core machine learning
accelerator targeting stencil and
tensor arithmetics.



CHI NoC and SerDes

On-chip high-speed network based
on multiple CHI cross points (XP).

Off-chip link based on SerDes.



Physical design by  Fraunhofer

Prototype board integration by 

EPAC: EPI Accelerator v1.5

GF22FDX, 27 mm², 0.3 Btr Tape out Mar 2023, Bring up Oct 2023

VEC tile

General purpose RISC-V CPU
Avispado Core (16 kI\$, 32 kD\$)
with dedicated VPU
Up to 256 DP element vector length



L2-HN tile

Distributed L2 cache (256 kB/slice) and
Coherence Home Node



VRP tile

General purpose RISC-V CPU
supporting variable precision
arithmetic up to 256 bit elements



STX tile

RISC-V many-core machine learning
accelerator targeting stencil and
tensor arithmetics.



CHI NoC and SerDes

On-chip high-speed network based
on multiple CHI cross points (XP).



Off-chip link based on SerDes.



Physical design by  Fraunhofer

Prototype board integration by  E4
COMPUTER
ENGINEERING

What's special in EPAC – VEC?

- The “Avispado” RISC-V core  silicon design and verification services
- The Vector Processing Unit (VPU)  **Barcelona
Supercomputing
Center**
Centro Nacional de Supercomputación

- 16 kB instruction cache
- 32 kB data cache
- Decodes v0.7, v1.0 vector extension
- Full hardware support for unaligned accesses
- Cache coherent (CHI)
- Vector memory accesses (vle, vlse, vlxe, vse, ...) processed by a dedicated queue (MIQ/LSU)



VPU with Long Vector Length (VL) support



— AVX512



← 512 bits per vector (8 DP elements)

arm — SVE



← Up to 2048 bits per vector (32 DP elements)

NEC

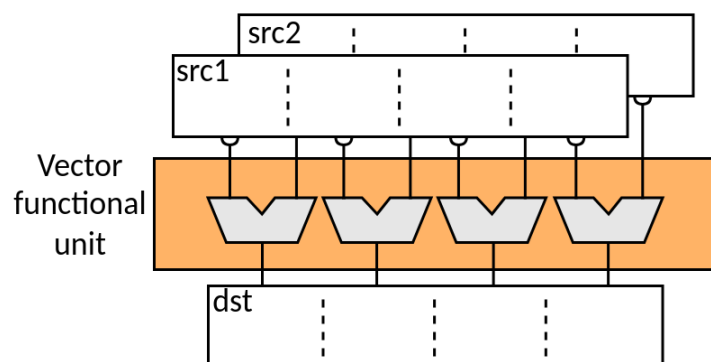


16384 bits per vector
(256 DP elements)



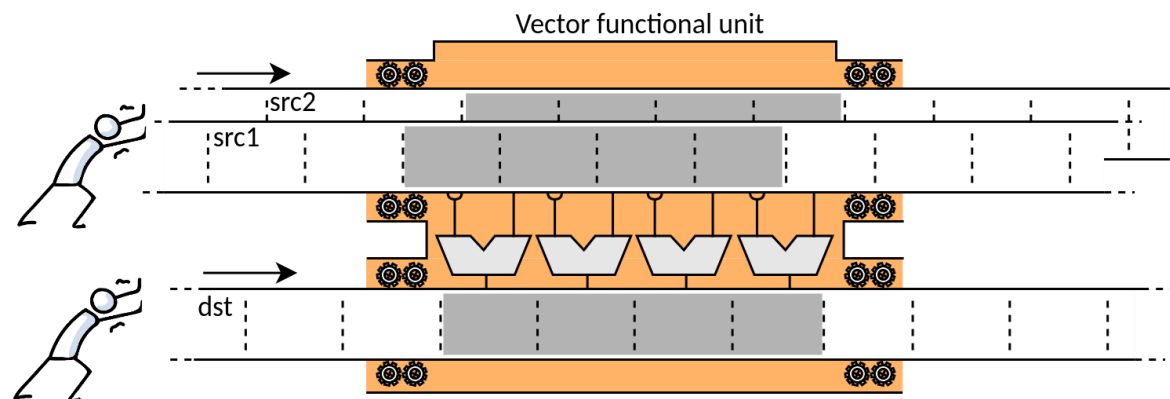
Short VL

- As many Functional Units as VL.
- Vector instructions executed in 1 cycle

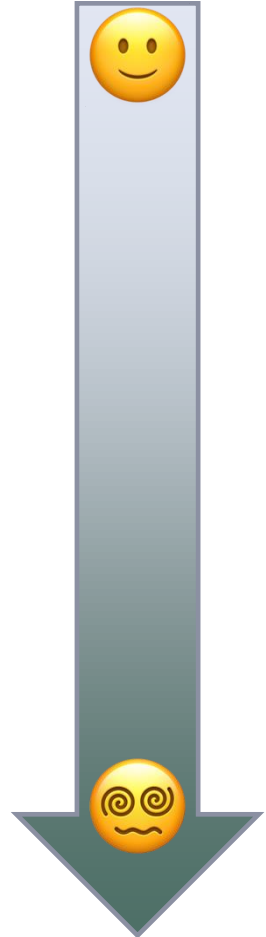


Long VL

- Cannot afford (area, power, cost) hundreds of Functional Units
- Vector instructions are executed on multiple cycles



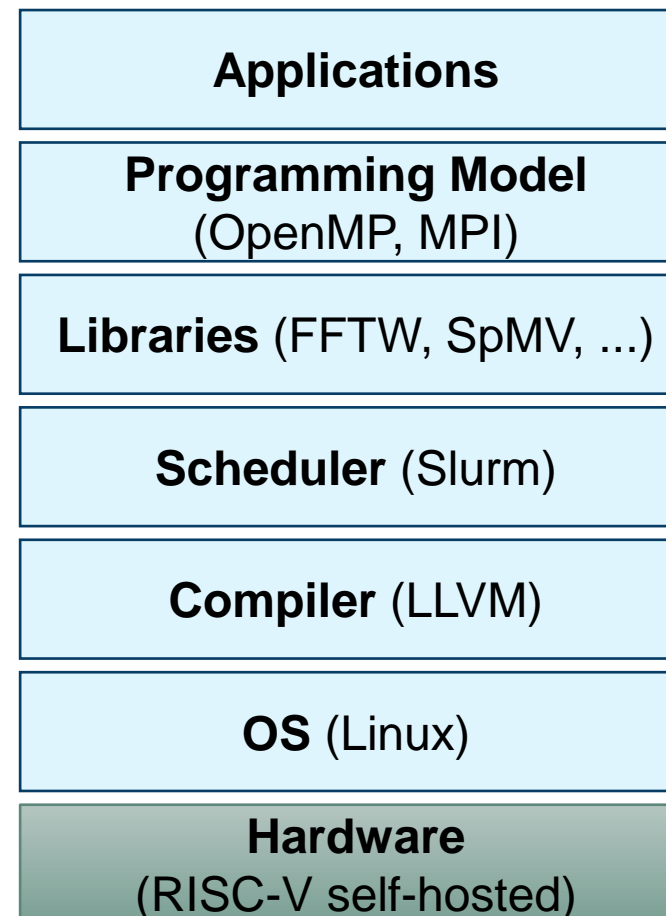
How do I program EPAC - VEC?



- **Autovectorization**
 - Leave it to the compiler
- **#pragma omp simd** (aka “Guided vectorization”)
 - Relies on vectorization capabilities of the compiler
 - Usually works but gets complicated if the code calls functions
 - Also usable in Fortran
- **C/C++/FORTRAN builtins** (aka “Intrinsics”)
 - Low-level mapping to the instructions
 - Allows embedding it into an existing C/C++ codebase
 - Allows relatively quick experimentation
- **Assembler**
 - Always a valid option but not the most pleasant

How do I use EPAC - VEC?

- Like a standard HPC system!
- Compile your code
 - We give you a compiler
- Link libraries
- Write/Submit a job script
 - SLURM
- Wait for the results
- Analyse execution traces and study how well your code is vectorized

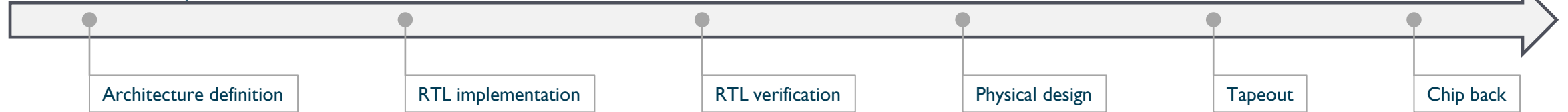


Why do you call it “Accelerator”?

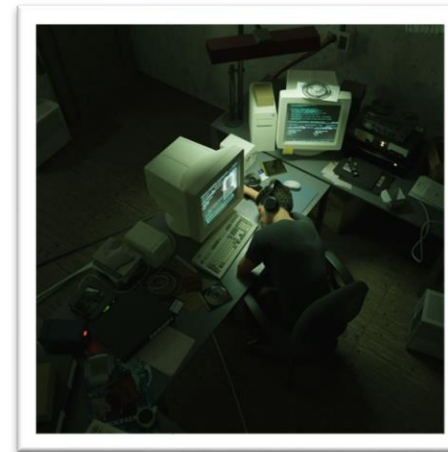
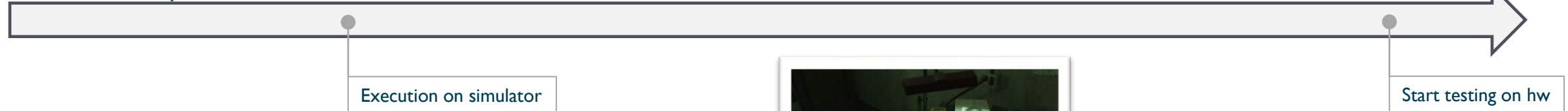
- Imagine you can boot Linux on a GPU:
Do you really think you would write CUDA kernels?
- EPAC supports both operational models
 - Self-hosted model (booting OS)
 - Host-device model (GPU-like)

Software development BEFORE hardware is ready

Hardware development



Software development



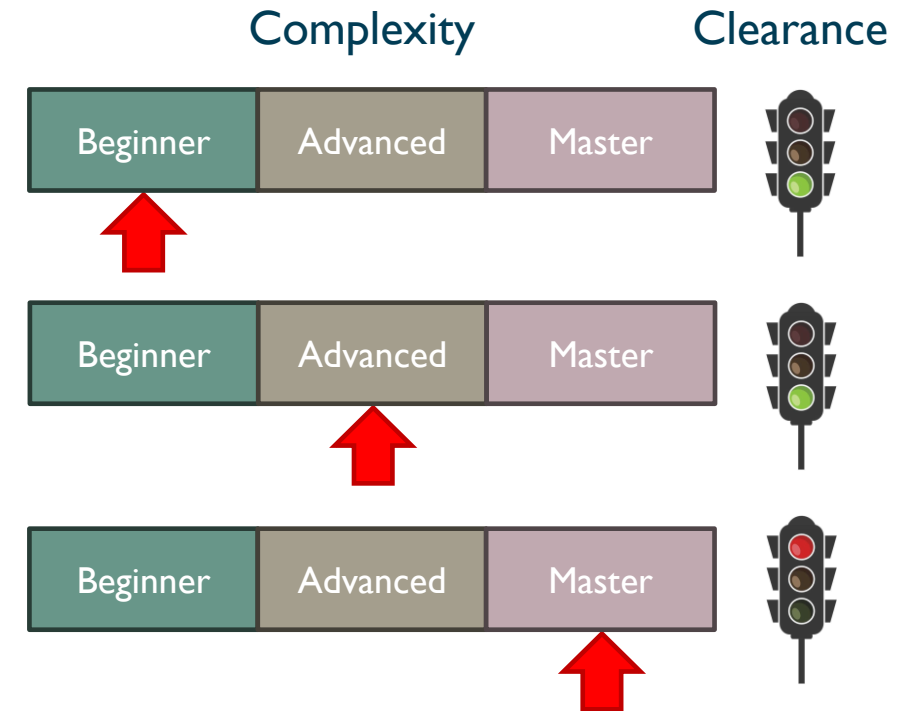
Wake up Neo...

Follow the Software Development Vehicles

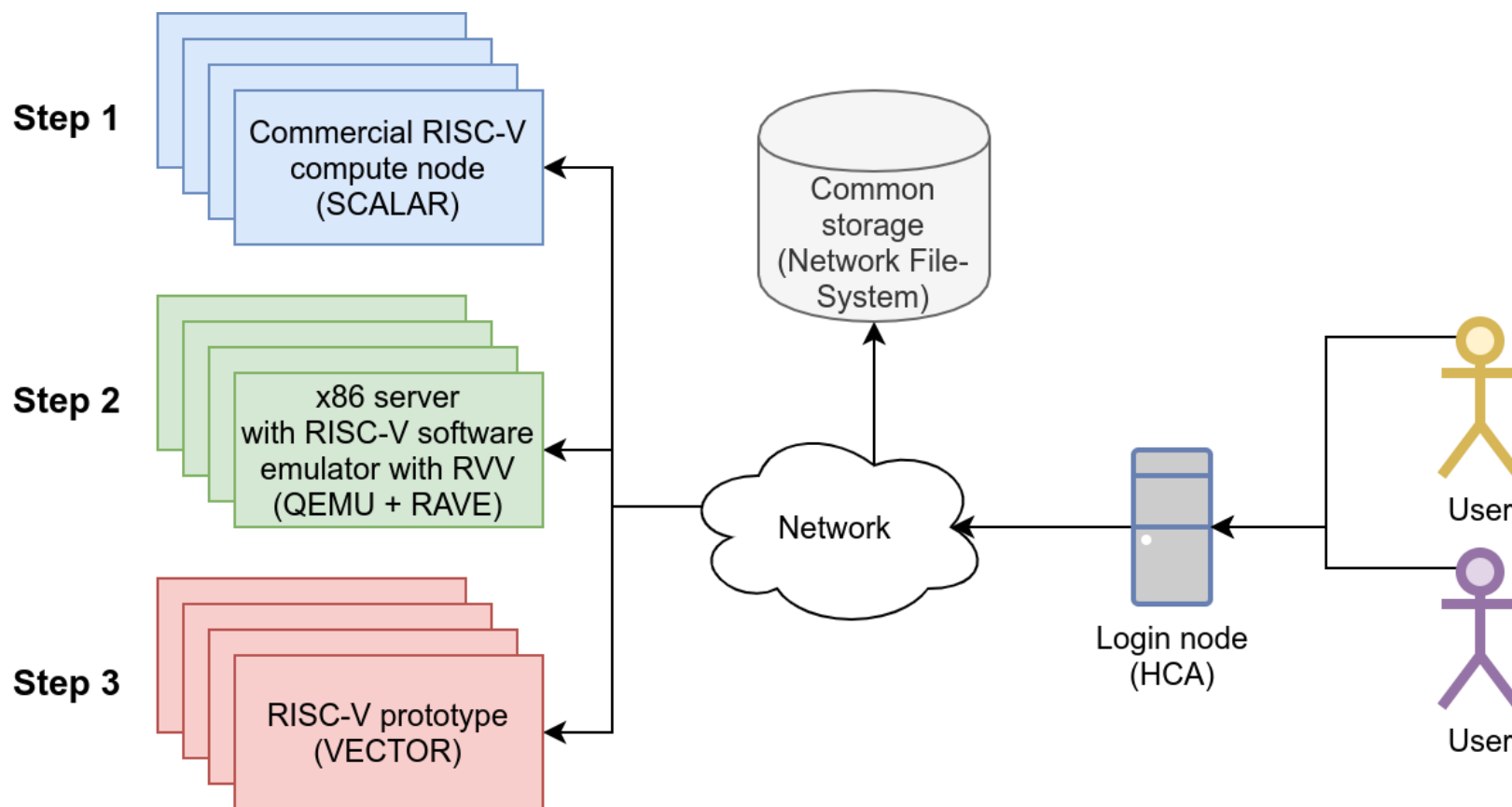


Software Development Vehicles (SDV)

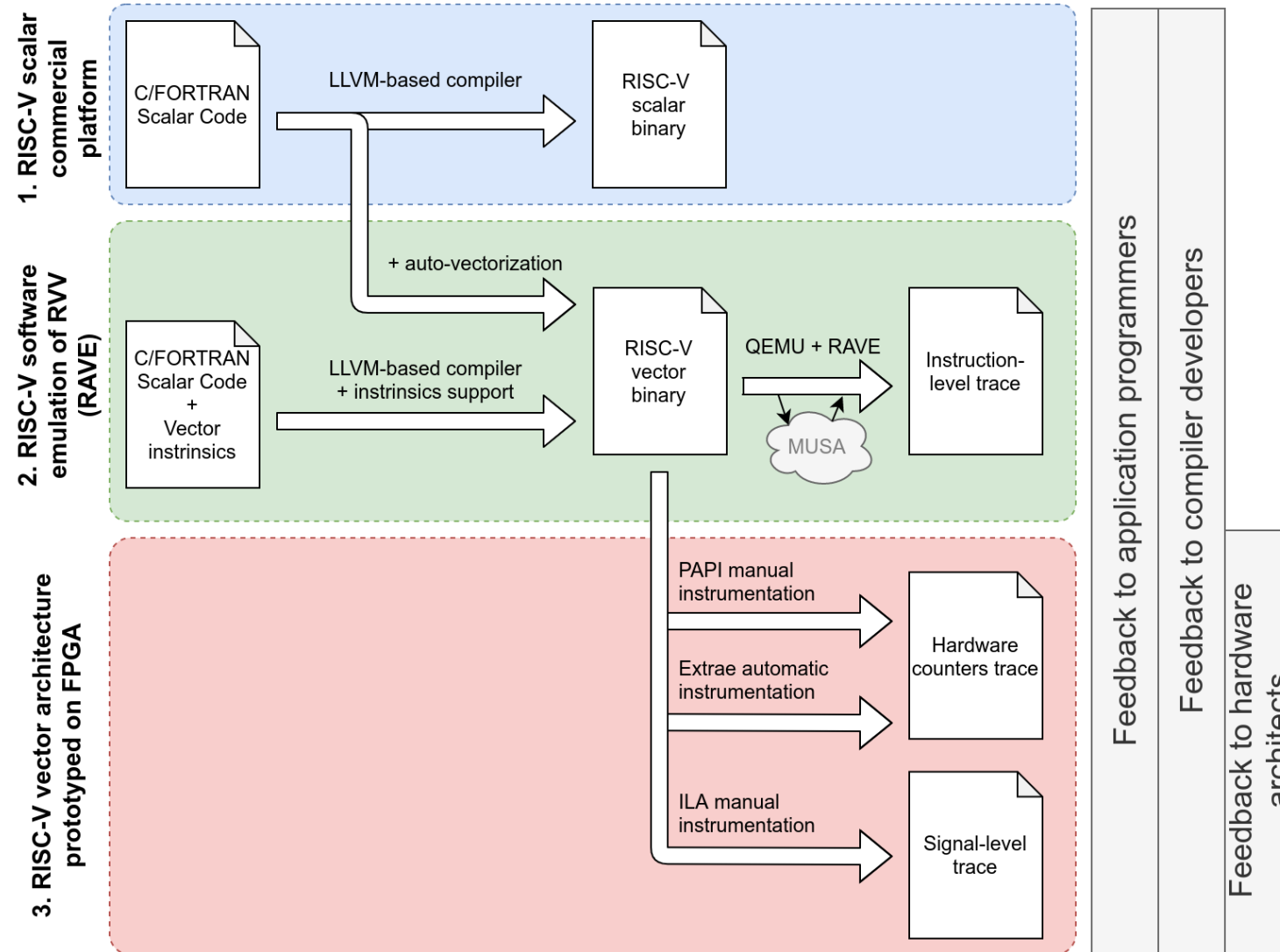
- 3 Steps:
 - **1st step:** Run in a commercial RISC-V platform (scalar CPU)
 - **2nd step:** RISC-V software emulation supporting RVV (RAVE)
 - **3rd step:** Run on VEC mapped into FPGA



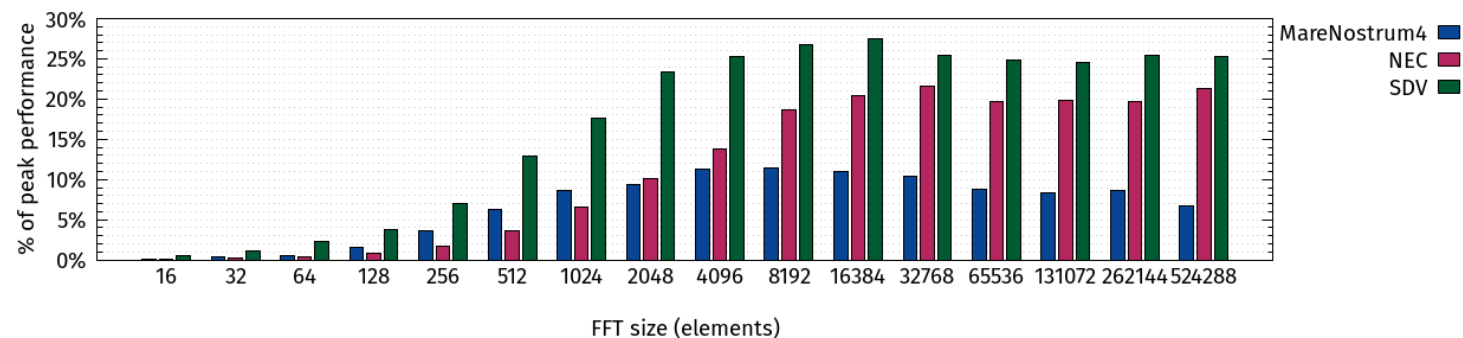
Software Development Vehicles (SDV)



Co-design with SDV



Real performance measurements



Vizcaino, Pablo, et al. "Acceleration with long vector architectures: Implementation and evaluation of the FFT kernel on NEC SX-Aurora and RISC-V vector extension." Concurrency and Computation: Practice and Experience 35.20 (2023): e7424.

SPMV (slowdown: time normalized to 0 extra latency)

Extra latency	0	32	64	128	256	512	768	1024
scalar	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
vl-8	1.00	1.17	1.15	1.10	1.10	1.07	1.05	1.05
vl-16	1.00	1.35	1.32	1.24	1.20	1.13	1.12	1.12
vl-32	1.00	1.71	1.63	1.50	1.42	1.29	1.26	1.26
vl-64	1.00	2.85	2.46	2.34	2.05	1.88	1.67	1.57
vl-128	1.00	4.83	3.86	3.57	3.06	2.72	2.33	2.34
vl-256	1.00	6.68	5.32	4.81	4.20	3.52	2.90	3.03
vl-512	1.00	8.78	6.70	6.26	5.16	4.32	3.61	3.39

BFS (slowdown: time normalized to 0 extra latency)

Extra latency	0	32	64	128	256	512	768	1024
scalar	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
vl-8	1.00	1.23	1.19	1.20	1.17	1.12	1.08	1.07
vl-16	1.00	1.47	1.37	1.37	1.33	1.32	1.27	1.19
vl-32	1.00	1.98	1.90	1.78	1.72	1.65	1.59	1.43
vl-64	1.00	2.94	2.53	2.53	2.44	2.38	2.19	1.97
vl-128	1.00	4.90	4.27	4.06	3.94	3.89	3.59	3.20
vl-256	1.00	6.83	5.72	5.56	5.97	5.20	4.84	4.50
vl-512	1.00	8.80	7.23	7.47	6.76	6.73	6.24	6.03

PR (slowdown: time normalized to 0 extra latency)

Extra latency	0	32	64	128	256	512	768	1024
scalar	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
vl-8	1.00	1.22	1.20	1.18	1.19	1.17	1.15	1.13
vl-16	1.00	1.44	1.40	1.38	1.37	1.34	1.32	1.27
vl-32	1.00	1.89	1.81	1.76	1.75	1.69	1.64	1.57
vl-64	1.00	2.79	2.63	2.53	2.50	2.40	2.30	2.20
vl-128	1.00	4.61	4.30	4.09	3.99	3.83	3.62	3.51
vl-256	1.00	6.44	6.04	5.66	5.48	5.22	4.95	4.70
vl-512	1.00	8.28	7.65	7.31	7.05	6.67	6.30	6.04

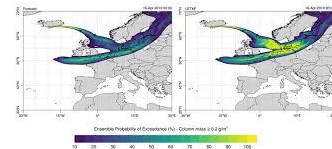
FFT (slowdown: time normalized to 0 extra latency)

Extra latency	0	32	64	128	256	512	768	1024
scalar	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
vl-8	1.00	1.11	1.03	1.03	1.02	1.02	1.02	1.01
vl-16	1.00	1.10	1.07	1.06	1.05	1.04	1.04	1.04
vl-32	1.00	1.31	1.20	1.17	1.12	1.09	1.09	1.09
vl-64	1.00	1.56	1.49	1.44	1.31	1.22	1.20	1.19
vl-128	1.00	2.20	2.23	1.88	1.74	1.56	1.47	1.49
vl-256	1.00	2.93	2.78	2.54	2.17	1.92	1.78	1.74
vl-512	1.00	3.60	3.59	3.23	2.59	2.28	2.05	2.03

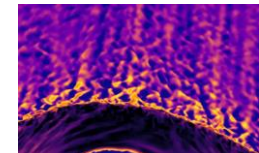


Vizcaino, Pablo, et al. "Short reasons for long vectors in HPC CPUs: a study based on RISC-V." Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis. 2023.

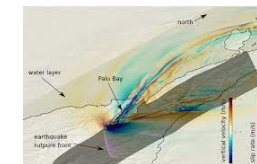
Real science



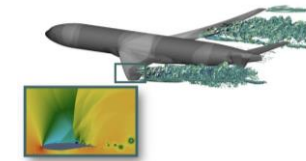
Fall3D



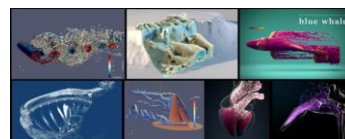
Vlasiator



SiesSol



FLEXI



Alya



Gene-X



References



- 📢 Webinar: Oct 10, 4:00 – 5:15 PM, RISC-V Technical Session
RAVE: RISC-V Analyzer of Vector Executions
- 📄 Mantovani, Filippo, et al. "Software Development Vehicles to enable extended and early co-design: a RISC-V and HPC case of study." International Conference on High Performance Computing. Cham: Springer Nature Switzerland, 2023. <https://arxiv.org/abs/2306.01797>
- 📄 Vizcaino, Pablo, et al. "Short reasons for long vectors in HPC CPUs: a study based on RISC-V." Proceedings of the SC'23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis. 2023. <https://arxiv.org/abs/2309.06865>
- 📄 Vizcaino, Pablo, et al. "RAVE: RISC-V Analyzer of Vector Executions, a QEMU tracing plugin." arXiv preprint arXiv:2409.13639 (2024). <https://arxiv.org/abs/2409.13639>
- 📄 <https://www.eetimes.com/examining-the-top-five-fallacies-about-risc-v/>
- 🎬 <https://www.youtube.com/watch?v=iFlcJFcOJKk>

EPI FORUM



EuroHPC
Joint Undertaking

PLATINUM SPONSORS



EVIDEN



GOLD SPONSORS



EPI FUNDING



This research has received funding from the European High Performance Computing Joint Undertaking (JU) under Framework Partnership Agreement No 800928 (European Processor Initiative) and Specific Grant Agreement No 101036168 (EPI SGA2). The JU receives support from the European Union's Horizon 2020 research and innovation programme and from Croatia, France, Germany, Greece, Italy, Netherlands, Portugal, Spain, Sweden, and Switzerland. The EPI-SGA2 project, PCI2022-132935 is also co-funded by MCIN/AEI /10.13039/501100011033 and by the UE NextGenerationEU/PRTR.

SPONSORED BY THE



Financé
par

