# SOftware research and development vehicles for New ARchitectures (SONAR)

The research activity of the SONAR group lays between the computer architecture (e.g. memory address translation, vector architectures) and high-performance computing (e.g. sparse linear algebra, parallel deep learning) areas.

## Summary

From the computer architecture perspective, we propose innovative solutions to maximize data reuse for fundamental hardware components like the register file, the processor front-end, or the cache hierarchy. Additionally, we propose novel high-performance computing approaches to increase the performance of sparse linear algebra and deep learning workloads on parallel architectures like many-core and vector processors. The SONAR group developes and maintains the MUltiscale Simulation Approach (SUMA) an accurate microarchitecture simulator targeting large-scale executions and long-vector architectures.

## Objectives

Numerical Libraries for parallel architectures:
The SONAR group is developing libraries for sparse linear algebra and recurrent neural networks. The group considers essential routines like the Sparse Matrix - Vector multiplication (SpMV), the Sparse Matrix - sparse Matrix (SpMM) product, or the Factorized Sparse Approximate Inverse (FSAI) preconditioner.

Hardware techniques to maximize data reuse:
The SONAR group is working on maximizing data reuse at key hardware components like the register file, the Translation Lookaside Buffer (TLB) or the instruction cache. Our approaches exploit unexplored sources of performance like exploiting data locality at the page table or the register file levels. The group targets high-performance computing architectures like vector processors or many-core architectures.

Micro-architecture simulation targeting long-vector architectures and large-scale executions:
We develop and maintain a microarchitecture model targeting long-vector and large-scale executions with a special focus on the RISC-V Vector extension. Our model uses a Paraver trace describing the dynamic stream of instructions of a parallel workload to drive a microarchitecture simulation. From this simulation, we enrich the trace by adding precise information in terms of instruction latency, memory hierarchy level where accesses are served, or commit cycle time. Our timing model keeps all information present in the first Paraver trace unaltered and adds additional timing and performance data. Please check the link below to learn how to download and use our RISC-V simulator in the context of the BSC RISC-V tools:

- https://repo.hca.bsc.es/gitlab/epi-public/risc-v-vector-simulation-environment/-/wikis/BSC-RISC%E2%80%90V-Vector-Toolchain