MEEP

MareNostrum Experimental Exascale Platform

Optimizations for very long and sparse vector operations on a RISC-V VPU : a work-in-progress

Gopinath Mahale,
Tejas Limbasiya, Muhammad Asad Aleem, Luis Plana,
Aleksandar Duricic, Alireza Monemi, Xabier Abancens Calvo,
Teresa Cervero and John D. Davis

Barcelona Supercomputing Center

May 25, 2023

The MEEP project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 946002. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Croatia, Turkey.

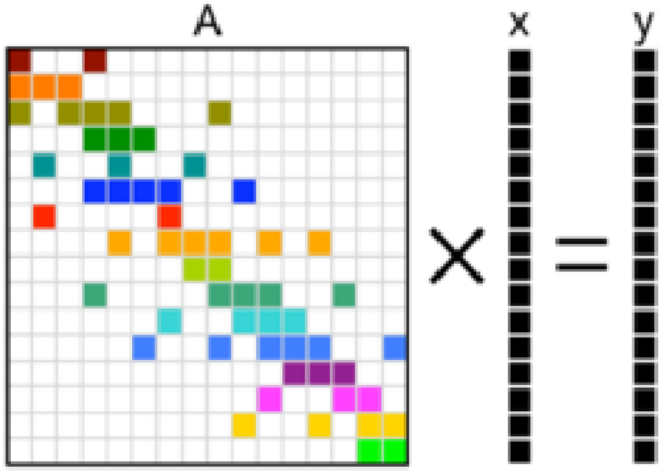
ISC High Performance
The HPC Event.

First International workshop on RISC-V for HPC

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Motivation

- Practical workloads in machine learning and scientific simulations are sparse in nature




```
void SpMV_ref(double *a, long *ia, long *ja, double *x, double *y, int nrows)
{
    int row, idx;
    for (row=0; row<nrows; row++) {
        double sum = 0.0;
        for (idx=ia[row];   idx<ia[row+1];    idx++) {
            sum += a[idx] * x[ ja[idx] ];
            }
        y[row] = sum;
        }
    }
}
```

- Vector memory accesses:
  - Sparse data vector access
  - non-unit stride dense vector access

} **Low-throughput** sparse memory accesses

- Solution: Large memory windows -> Optimize main memory accesses by reordering
  -> **Higher MLP**

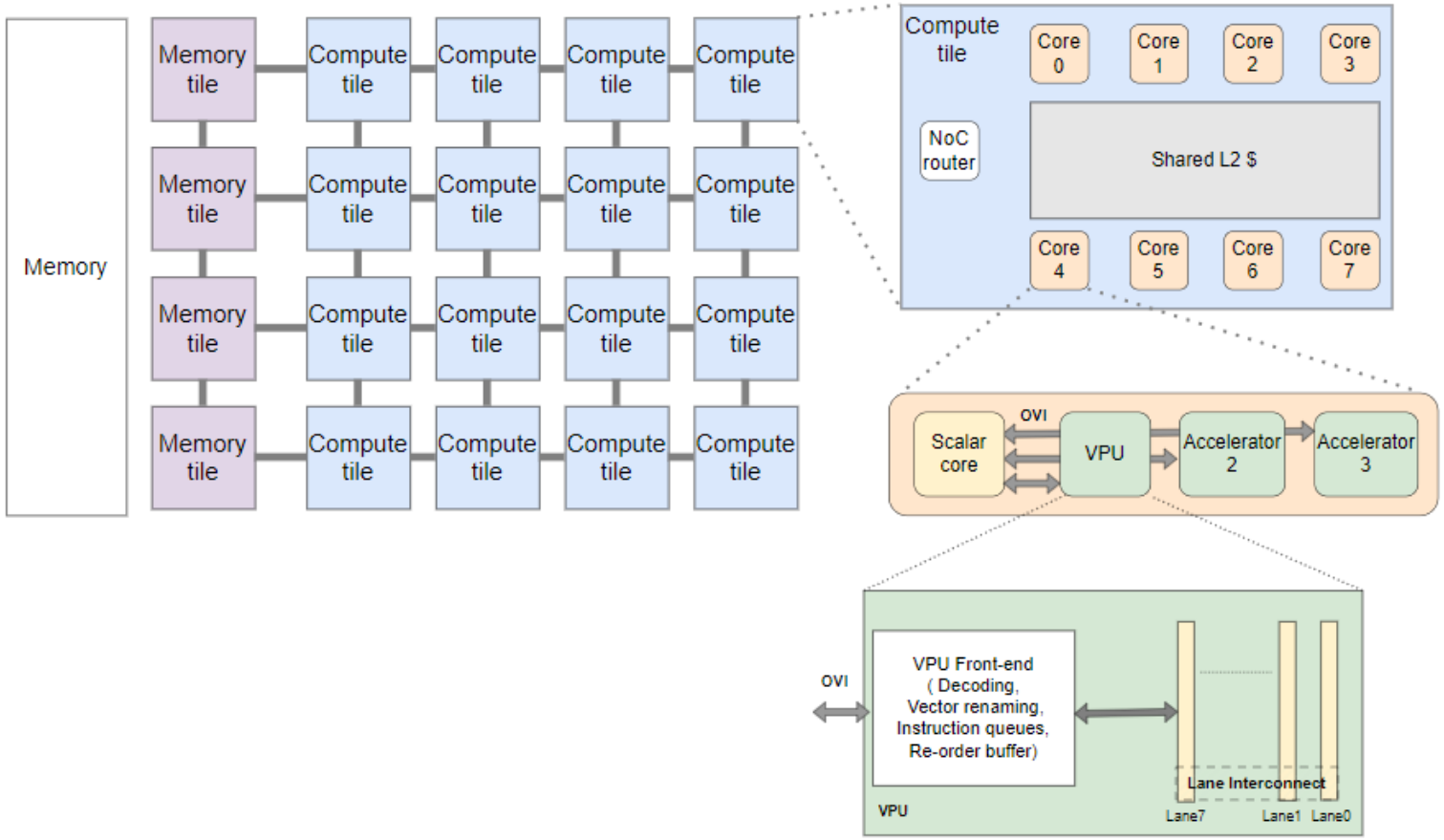
- Is data caching beneficial for long sparse vector operations?

Image source: https://shorturl.at/bmADH

Accelerated Compute and Memory Engine (ACME)

- Two modes of operation
  1. Classic-mode
  2. ACME-mode

- **Memory tile**
  - Memory path bypassing L1 and L2 caches
  - Network-efficient indexed loads
  - Non-speculative pre-fetch of vector data
  - Handles L2 cache miss, L1 TLB miss

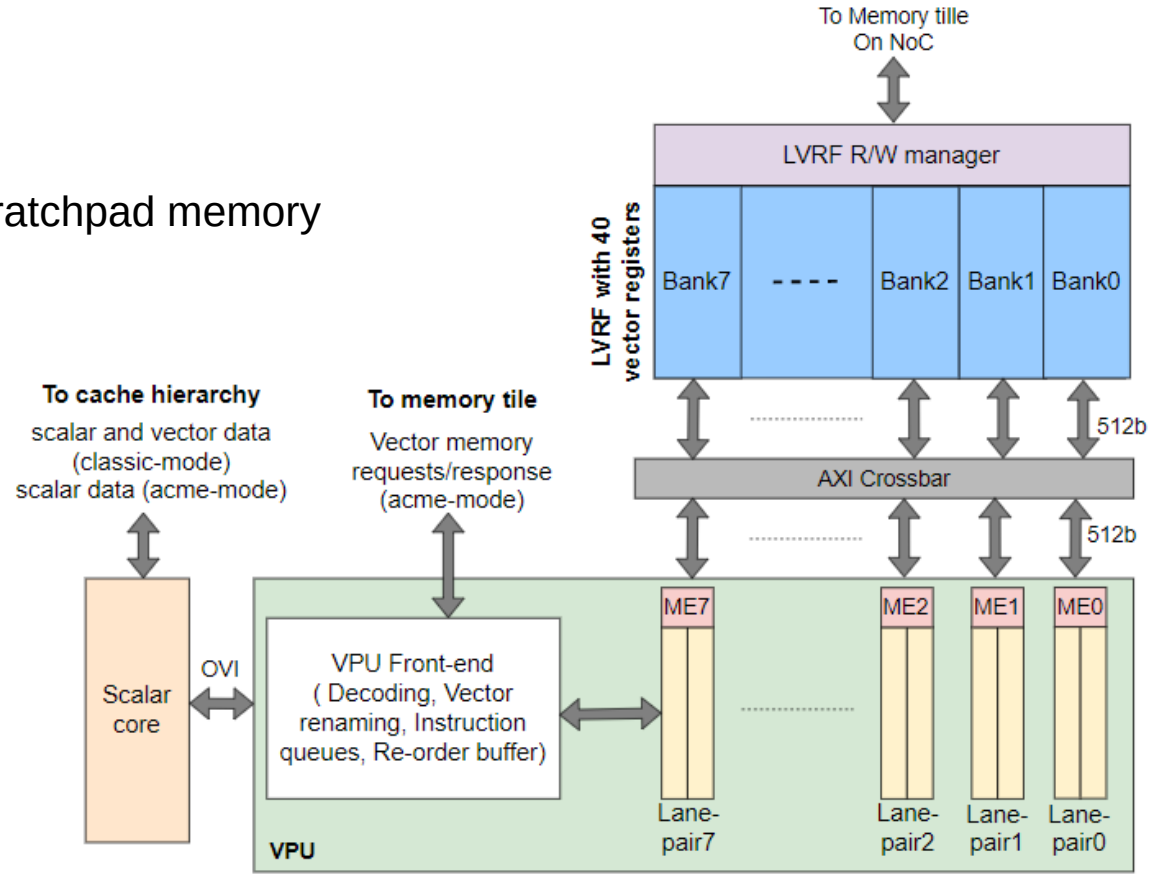


The baseline VPU

VPU: Vector processor unit
OVI: Open Vector Interface (from Semidynamics)

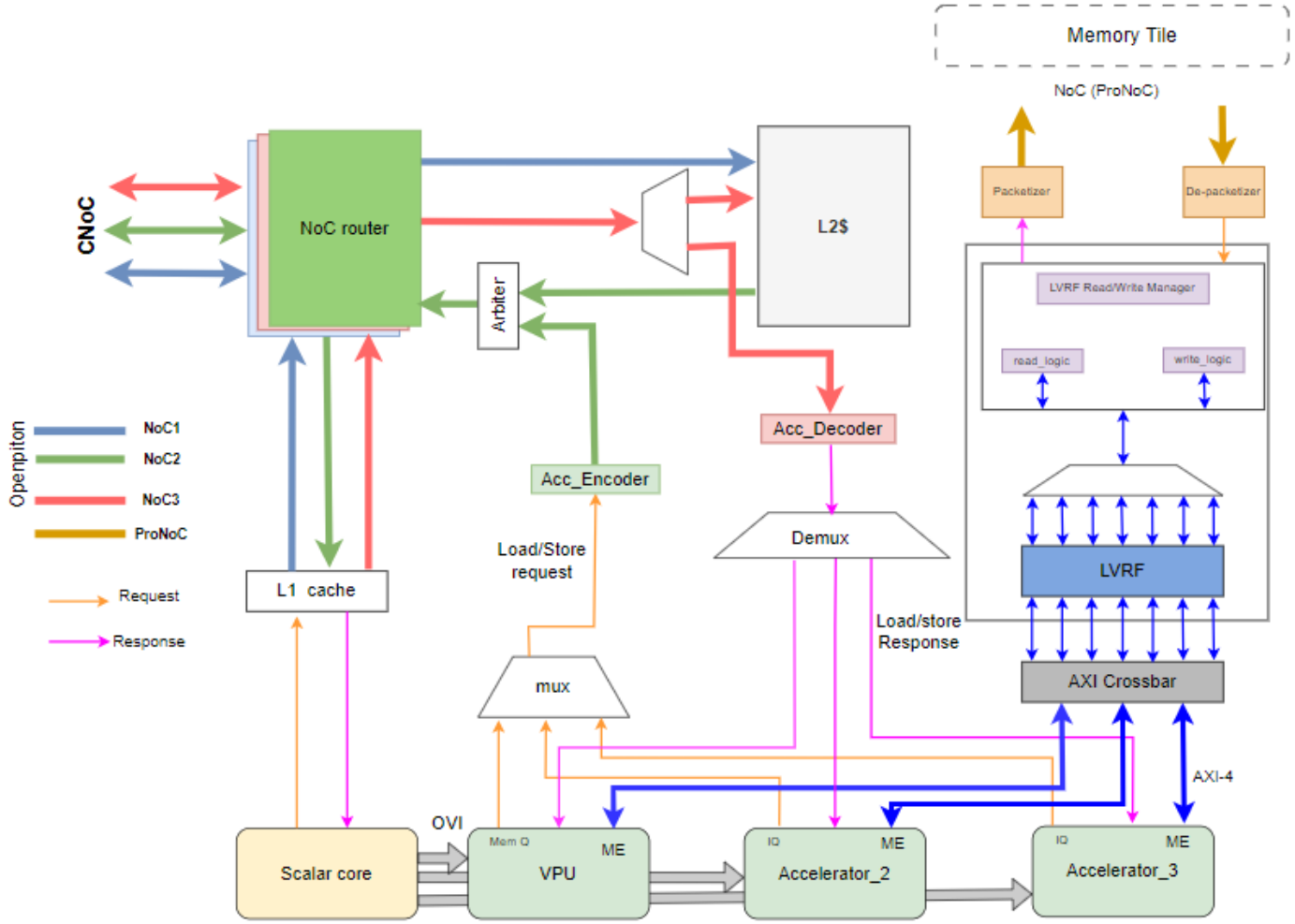
The proposed VPU and instruction execution

- Features of the proposed VPU:
  - RVV 0.7.1
  - Long vector register file (LVRF) on a scratchpad memory
  - Vector lane pairs
  - Two modes of operation
    1. Classic mode
    2. ACME mode
  - Modifications to the compute path to support hardware data strip-mining
  - Interfaces to the memory tile for memory req/resp



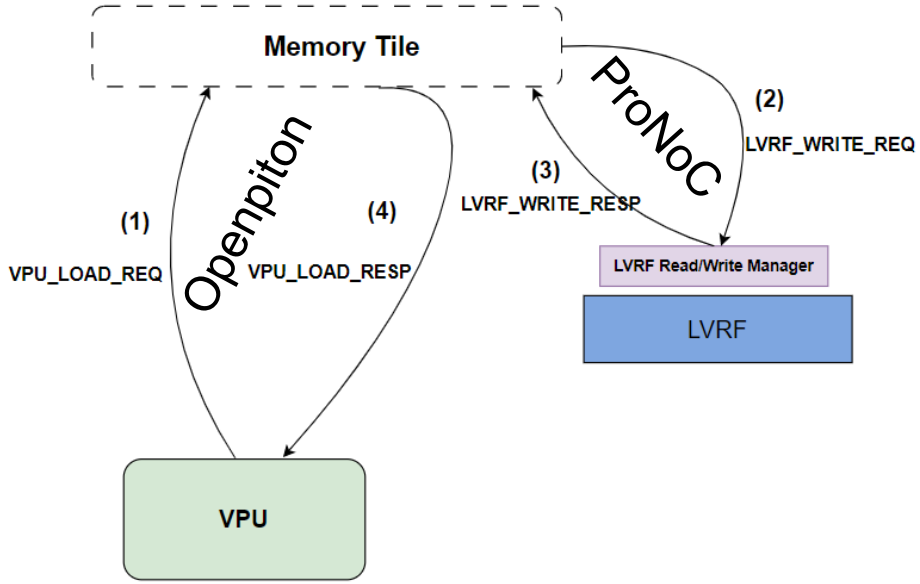
ME: Microenignes
LVRF: Long Vector Register File
OVI: Open Vector Interface (from Semidynamics)

FPGA implementation: Definition of new interfaces, message types



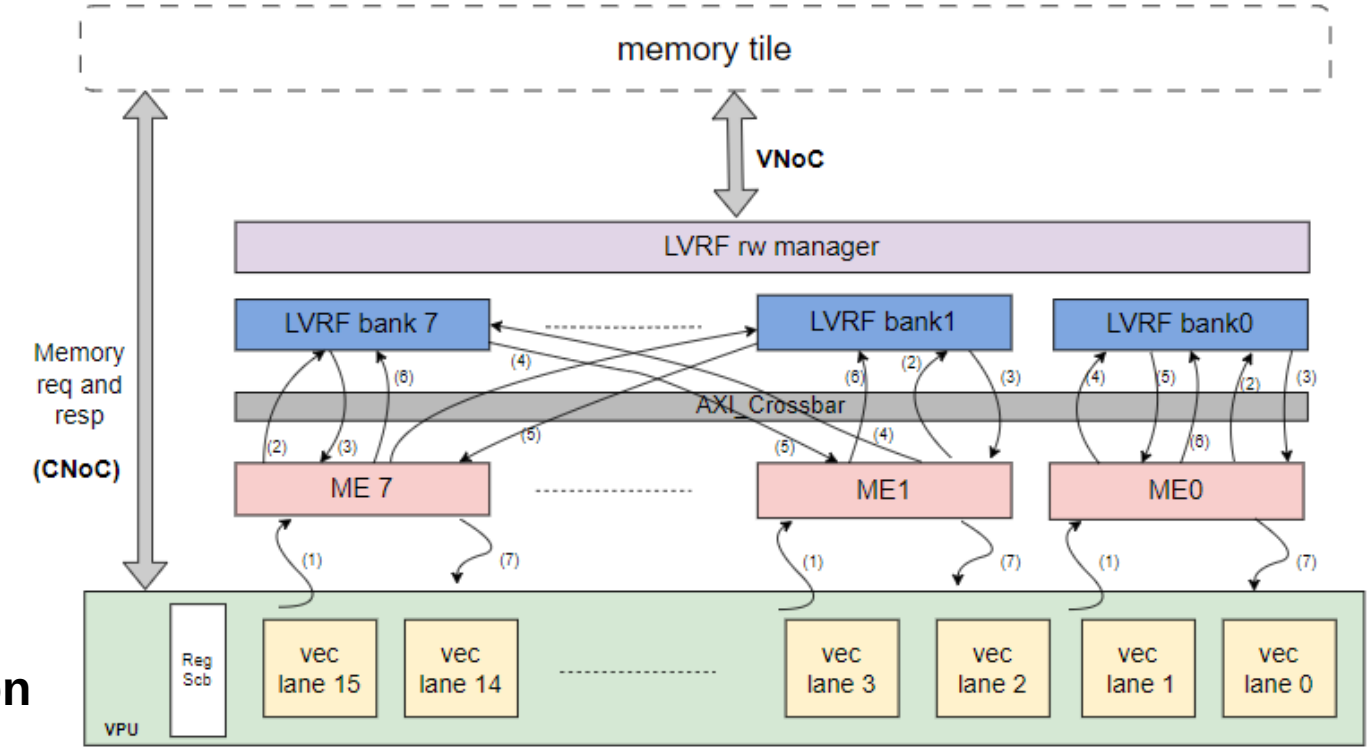
Communication between

- Accelerators and memory tile (over CNoC) : **Openpiton NoC**
- LVRF and memory tile (over VNoC) : **Prototype-network-on-chip (ProNoC)**



RISC-V vector instruction execution in acme-mode

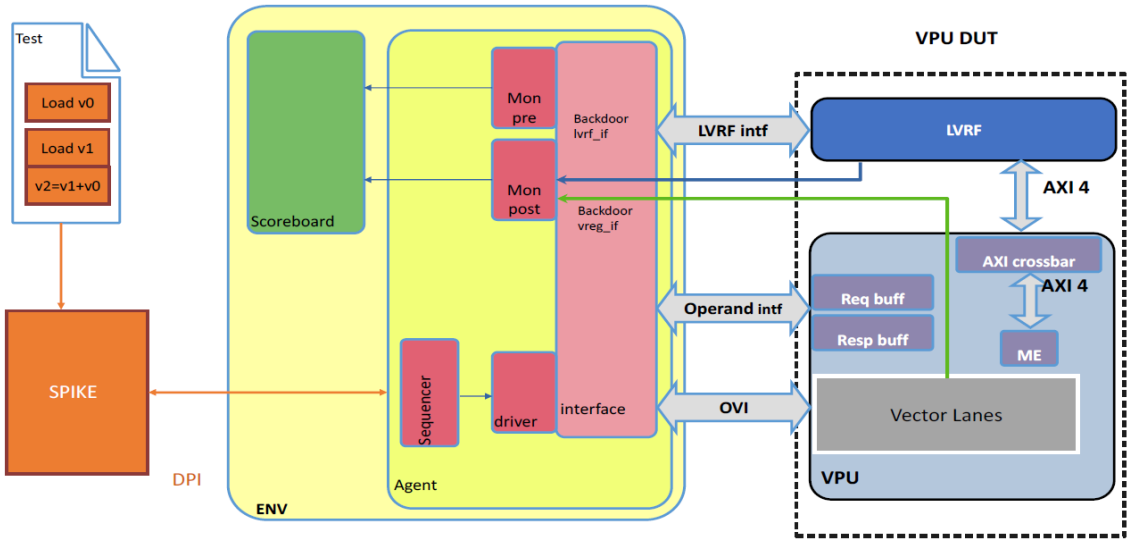
- **Vector memory instructions**
  - non-indexed vector loads/stores
  - Indexed vector loads/stores
- **Vector Non-memory instructions**
  - arithmetic instructions
  - gather, slide instructions
- **Vector masked instruction execution**
  - arithmetic instruction
  - memory instruction



(1) Configuration of MEs by VPU
(2) MEs request 512 bits of index vector fragment from LVRF
(3) MEs receive the index vector fragment
(4) A new request for every index is generated (for example element width of 64 generates 8 reads from LVRF)
(5) Receive 512 bit vector data containing indexed value data from LVRF. Separate the indexed value.
Repeat step 4 and step 5 multiple times based on element width
(6) Write the read indexed vector elements packed as a cache line to LVRF
Go to step 2 until the vector length is covered
(7) VPU is notified about the instruction completion, and the instruction is committed

The scheme for vrgather

Verification environment and preliminary results



# Kernel	Characteristics	classic-mode	acme-mode
MatMul	Dense matrix multiplication	PASS	PASS
SpMV	Sparse matrix vector multiplication	PASS	PASS
Somier	Dense linear algebra kernel	PASS	PASS
Axpy	Dense Axpy	PASS	PASS
FFTW	Fast Fourier transform	PASS	In progress

Simulation results on bench kernels from RISC-V vectorized benchmark suite

FPGA synthesis results:

# lanes	Total LUTs	Logic LUTs	LUTRAMs	SRLs	FFs	RAMB36	RAMB18	URAM	DSP Blocks
2 lanes	100412	99208	1204	0	89881	112	16	0	22
4 lanes	194604	192366	2236	2	173484	128	0	0	44
8 lanes	385882	381574	4304	4	340782	128	0	0	88
16 lanes	768877	760429	8440	8	679515	128	0	0	176
Available	1303680	1303680	19934		868528	2016	4032	960	9024

Resource utilization for an integration of VPU, ME and LVRF on Xilinx Alveo U55C FPGA

Summary

- ACME architecture handles very long and sparse vector operations
  - Increases dynamic memory window -> improving MLP
  - Reduces transfer of parasitic data
  - Non-speculatively pre-fetches vector data

- The proposed RISC-V VPU is equipped with
  - long vector operation support -> larger vector registers
  - a memory path that bypasses L1, L2 cache

- The VPU is integrated in the Openpiton framework, with dedicated memory req/resp message interfaces to the memory tile

- Schemes to execute different categories of RISC-V vector instructions on the VPU are presented

MEEP

MareNostrum Experimental Exascale Platform

Thank you!

gopinath.mahale@bsc.es

The MEEP project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 946002. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Spain, Croatia, Turkey.