Scaling Innovation

ParaStation MPI

MPICH BoF · SC21 · Virtual
Nov 17, 2021

Thomas Moschny
CTO, ParTec AG

ParTec
Modular Supercomputing

ParaStation Modulo

- ParaStation ClusterTools
  - *Tools for provisioning and management*
- ParaStation HealthChecker & TicketSuite
  - *Automated error detection & error handling*
  - *Ensuring integrity of the computing environment*
  - *Keeping track of issues*
  - *Powerful analysis tools*
- ParaStation MPI & Process Management
  - *Runtime environment specifically tuned to the largest distributed memory supercomputers*
  - *Scalable & mature software setup*
  - *Full batch system integration (e.g., Slurm)*

ParaStation
MODULO

Maximize job throughput –
Minimize administration effort

ParaStation Modulo

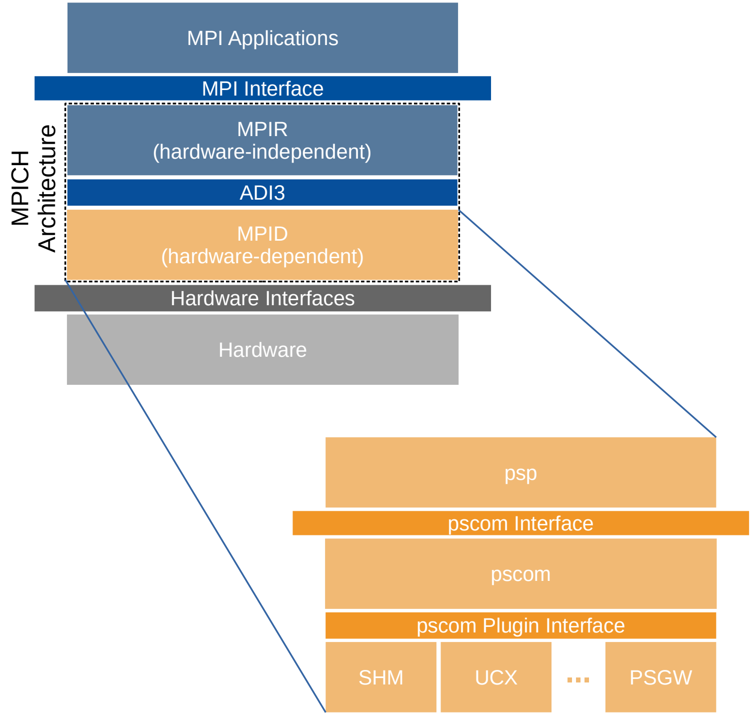
- ParaStation ClusterTools
  - *Tools for provisioning and management*
- ParaStation HealthChecker & TicketSuite
  - *Automated error detection & error handling*
  - *Ensuring integrity of the computing environment*
  - *Keeping track of issues*
  - *Powerful analysis tools*
- ParaStation MPI & Process Management
  - *Runtime environment specifically tuned to the largest distributed memory supercomputers*
  - *Scalable & mature software setup*
  - *Full batch system integration (e.g., Slurm)*

ParaStation MODULO

Maximize job throughput –
Minimize administration effort

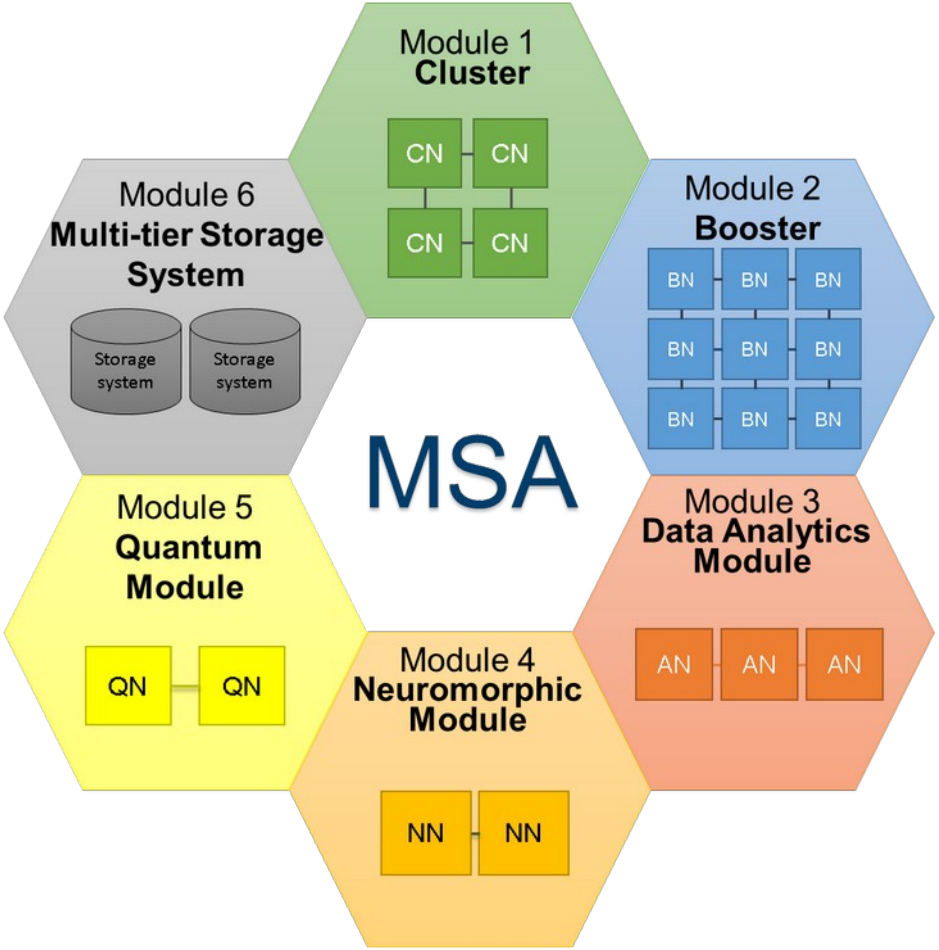
ParaStation MPI Architecture

- Based on MPICH 3.4.2 (MPI-3.1 compliant)
  - *Supports MPICH tools (tracing, debugging, …)*
  - *MPICH layers beneath ADI3 are replaced by ParaStation PSP Device*
  - *Powered by pscom low-level communication library*
  - *Maintains MPICH ABI compatibility*
- Support for various transports / protocols via pscom plugins
  - *Support for InfiniBand, Omni-Path, Extoll, …*
  - *Multiple transports / plugins can be used concurrently*
  - *Gateway capability via PSGW plugin to bridge transparently between any pair of networks supported by the pscom*
  - *CUDA awareness for all transports / CUDA optimization via GPUDirect for UCX, and Extoll*
- Proven to scale up to ~3,500 nodes and ~140,000 processes per job



Modular Supercomputing Architecture

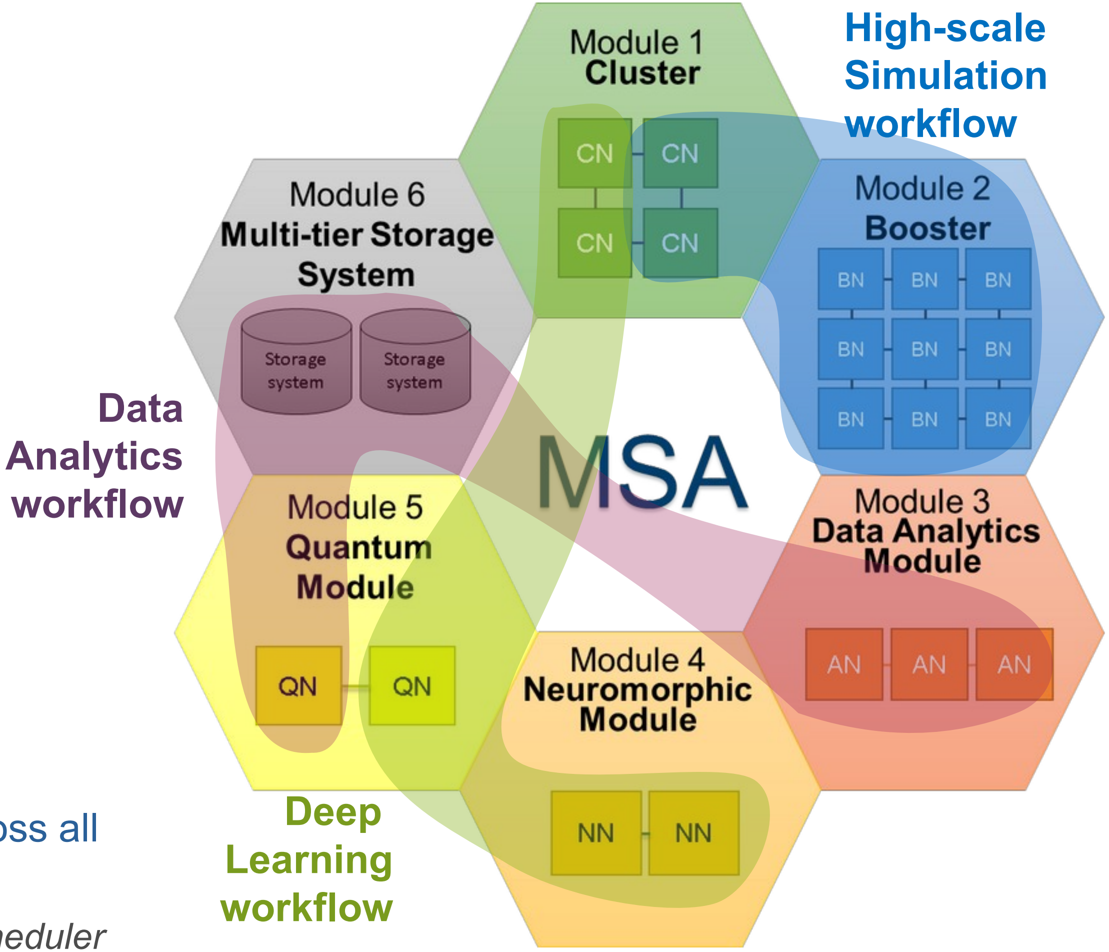
- Generalization of the Cluster-Booster Concept
  - *Composability of heterogeneous resources*
  - *Effective resource-sharing*
- Any number of (specialized) modules possible
  - *Cost-effective scaling*
  - *Extensibility of existing modular systems*
- Fit application diversity
  - *Large-scale simulations*
  - *Data analytics*
  - *Machine/Deep Learning, AI*
  - *Hybrid Quantum Workloads*
- Achieve leading scalability & energy efficiency → Exascale
- Unified SW environment to run applications across all modules
  - *ParaStation Modulo providing a Slurm-based Scheduler*



ParTec
Modular Supercomputing

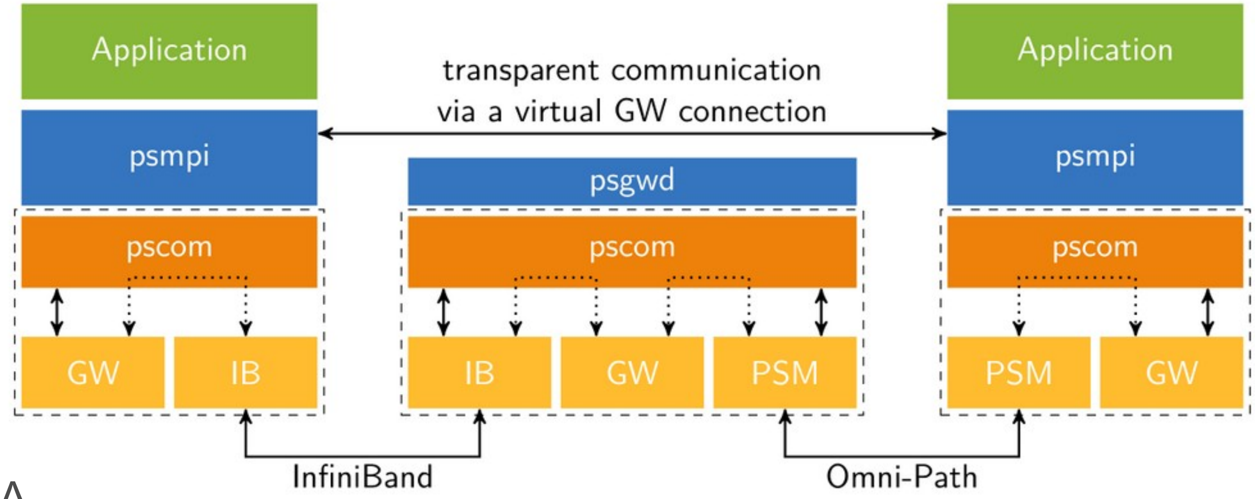
Modular Supercomputing Architecture

- Generalization of the Cluster-Booster Concept
  - *Composability of heterogeneous resources*
  - *Effective resource-sharing*
- Any number of (specialized) modules possible
  - *Cost-effective scaling*
  - *Extensibility of existing modular systems*
- Fit application diversity
  - *Large-scale simulations*
  - *Data analytics*
  - *Machine/Deep Learning, AI*
  - *Hybrid Quantum Workloads*
- Achieve leading scalability & energy efficiency → Exascale
- Unified SW environment to run applications across all modules
  - *ParaStation Modulo providing a Slurm-based Scheduler*



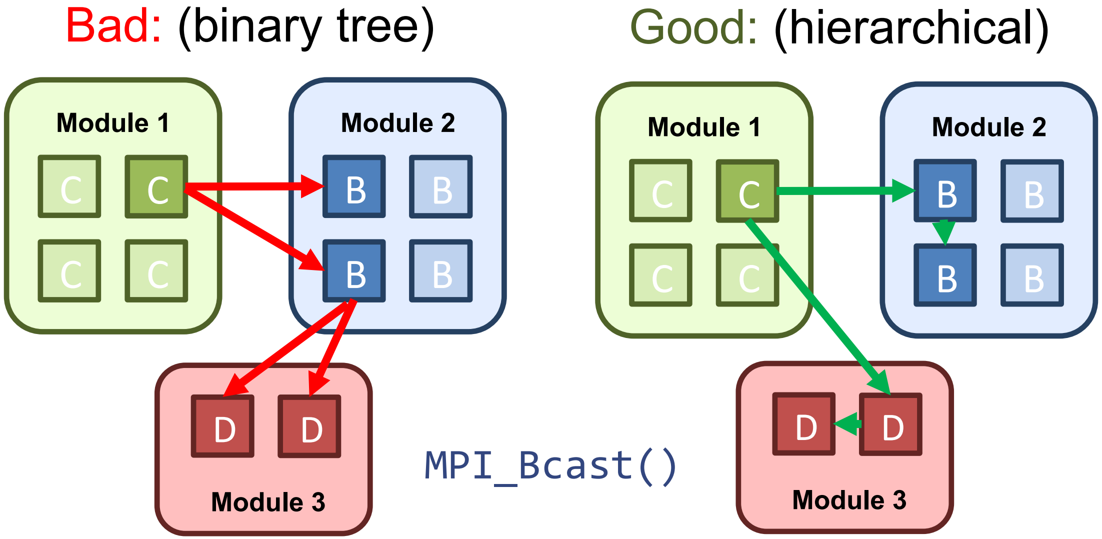
ParaStation MPI Network Bridging

- Two processes communicate through a gateway if they are not directly connected by a high-speed network (e.g., InfiniBand, OPA, Extoll, BXI, …)
- High-speed connections between processes and gateway daemons
- Static routing to choose a common gateway
- Virtual connection between both processes through the gateway, fully transparent for the application
- Virtual connections are multiplexed through gateway connections
- Implemented first for the JURECA Cluster-Booster System: Bridging between Mellanox EDR and Intel Omni-Path



MSA Awareness

- Modularity-aware MPI Collectives
  - *Optimized patterns for collectives that take the topology of the MSA system into account*
  - *Assumption: Inter-module communication is the bottleneck*
  - *Dynamic updates of the communication patterns supported, e.g., for malleable jobs (experimental)*



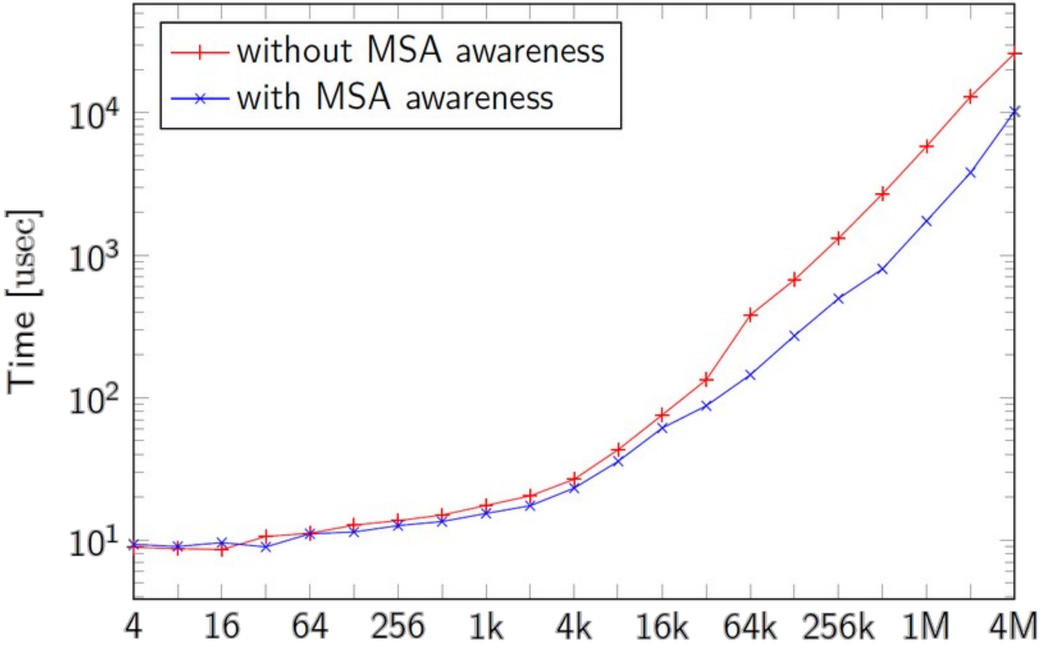
Hierarchical Collectives

- General rules used here to optimize collectives
  1) *First, do all module-internal gathering and/or reduction operations (if required)*
  2) *Second, conduct the inter-module operation with a single process per module*
  3) *Finally, perform a strict module-local distribution of the data*
- Multi-level hierarchy awareness
  - *Apply this set of rules* recursively*, i.e., module level, node level, etc.*
- Corresponding environment variables in ParaStation MPI
  - `PSP_MSA_AWARENESS = 1`
  - `PSP_MSA_AWARE_COLLOPS=1`
  - `PSP_SMP_AWARE_COLLOPS=1`

Performance Improvement

- Performance heavily depends on the concrete settings, i.e.:
  - Number of processes / gateway nodes
  - Distribution of the ranks in the communicator
  - Message sizes (and hence the collective communication pattern)
- Currently supported collectives
  - `MPI_Bcast / MPI_Ibcast`
  - `MPI_Reduce / MPI_Ireduce`
  - `MPI_Allreduce / MPI_Iallreduce`
  - `MPI_Scan / MPI_Iscan`
  - `MPI_Barrier`

IMB MPI Benchmarks: Allreduce with 8 (CN) + 8 (DAM-EXT) nodes, 8 procs per node, and 1 Gateway (GW) node on DEEP-EST prototype



ParTec
Modular Supercomputing

MSA-aware API Extensions

- Means for adapting modularity explicitly on the application level
- API extensions by ParaStation MPI for querying the topology information
  - *Query the module ID via the* `MPI_INFO_ENV` *object*

```
MPI_Info_get(MPI_INFO_ENV, "msa_module_id", …, value, …);
```

  - *New split type for creating communicators matching the MSA topology*

```
MPI_Comm_split_type(oldcomm, MPIX_COMM_TYPE_MODULE, …, &newcomm);
```

Detecting CUDA-awareness

- Support for the APIs introduced by OpenMPI (via `mpi-ext.h`)
  - *Compile-time macro*

```
#if defined(MPIX_CUDA_AWARE_SUPPORT) && MPIX_CUDA_AWARE_SUPPORT
printf("The MPI library is CUDA-aware\n");
#endif
```

  - *Runtime function*

```
if (MPIX_Query_cuda_support())
    printf("The MPI library is CUDA-aware\n");
```

- Additionally query the CUDA awareness via `MPI_INFO_ENV` (ParaStation MPI only!)

```
MPI_Info_get(MPI_INFO_ENV, "cuda_aware", sizeof(is_cuda_aware)-1,
             is_cuda_aware, &api_available);
```

Statistical Analysis

- Generate message size histogram by setting the `PSP_HISTOGRAM` environment variable
- Influence histogram via

`PSP_HISTOGRAM_MIN`	Lower message size limit
`PSP_HISTOGRAM_MAX`	Upper message size limit
`PSP_HISTOGRAM_SHIFT`	Bucket width

- Limit the analysis to a certain connection type (e.g., inter-module gateway traffic) by setting `PSP_HISTOGRAM_CONTYPE` accordingly

JUWELS – A Modular Supercomputer

© Forschungszentrum Jülich

© Forschungszentrum Jülich/Ralf-Uwe Limbach

JÜLICH Forschungszentrum | JÜLICH SUPERCOMPUTING CENTRE

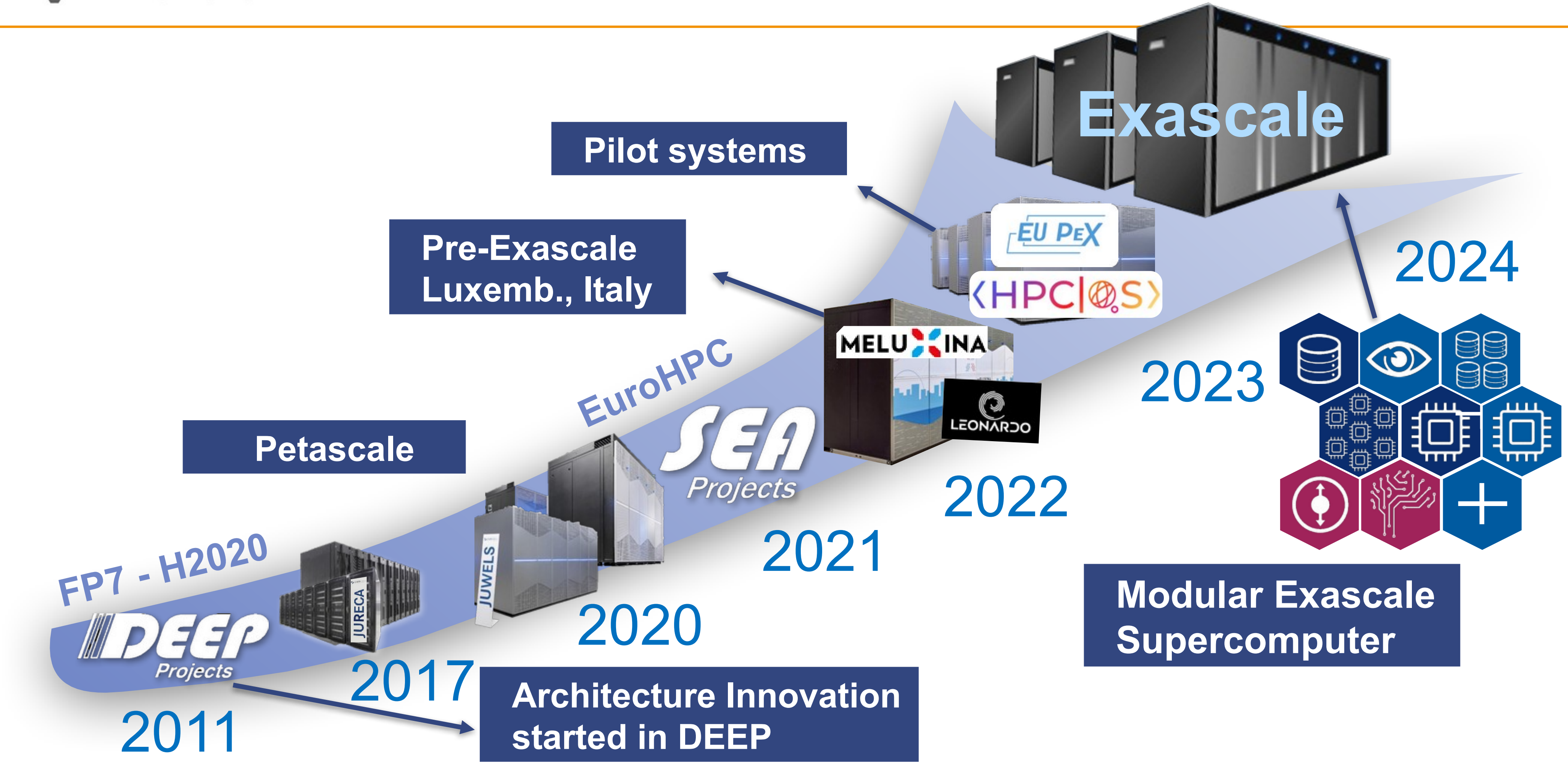
- 12 PFlop/s peak
- #23 on Top500 list (June 2018)
- 2575 nodes (Bull Sequana X1000)
- Intel Xeon Platinum 8168 / Gold 6148
- Mellanox EDR, ParaStation MPI

TOP 500 The List.

- GPU-accelerated module, 70 PFlop/s peak
- #7 on Top500, #3 on Green500 (Nov. 2020)
- 936 nodes (Bull Sequana XH2000)
- 4x NVIDIA A100 GPUs per node
- Quad-rail Mellanox HDR200, ParaStation MPI

Operated as one Modular System with ParaStation Modulo and Slurm

Modular Supercomputing to Exascale



Pilot systems

Pre-Exascale
Luxemb., Italy

Petascale

Architecture Innovation
started in DEEP

Modular Exascale
Supercomputer

Thank you for your attention!

Questions?

{clauss, moschny, pickartz}@par-tec.com