XPRESS: eXascale Programming Environment and System Software

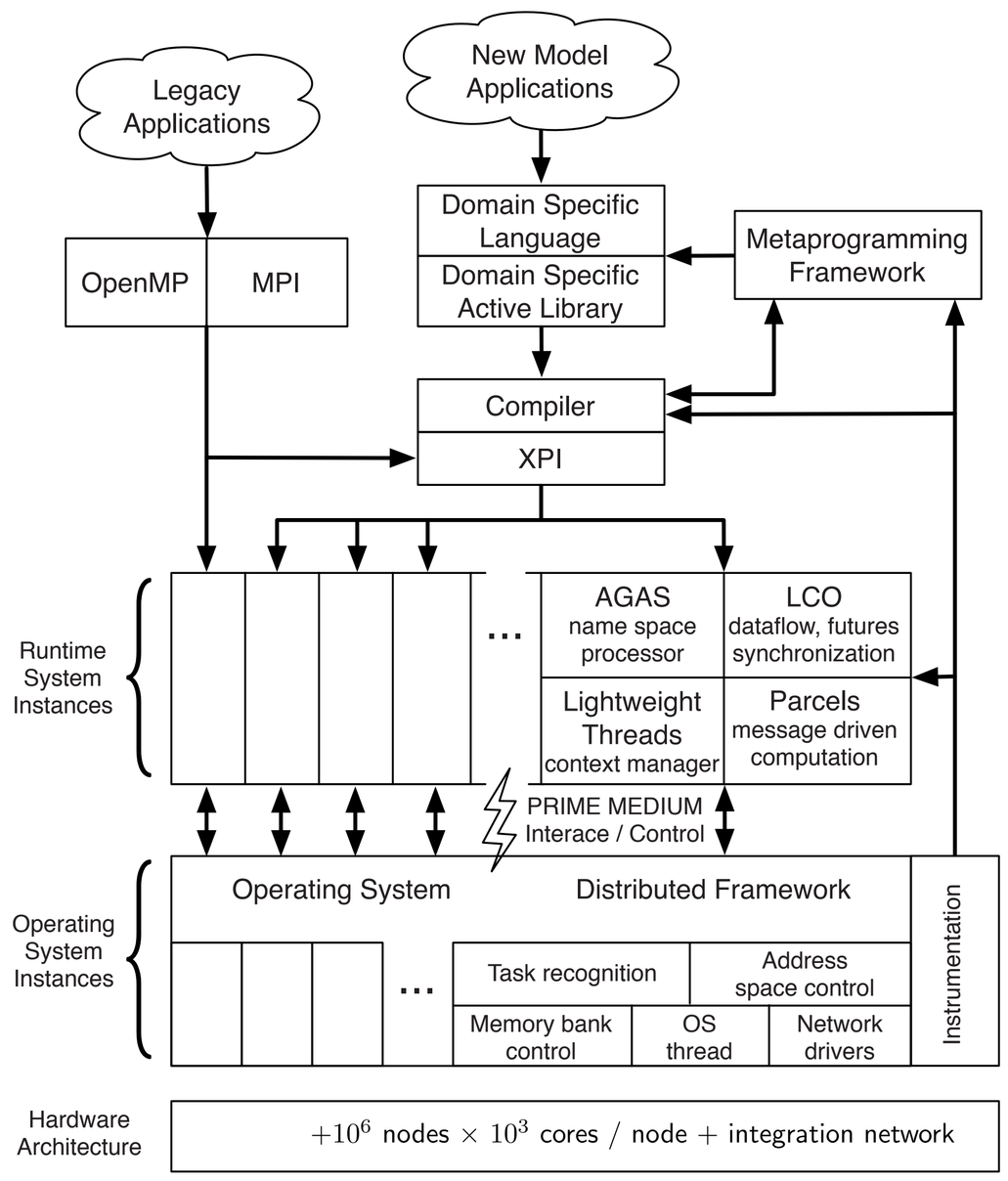
Goal

The XPRESS Project is one of four major projects of the DOE Office of Science ASCR X-stack Program initiated in September, 2012. The purpose of XPRESS is to devise an innovative system software stack to enable practical and useful exascale computing around the end of the decade with near-term contributions to efficient and scalable operation of trans-Petaflops performance systems in the next two to three years; both for DOE mission-critical applications. To this end, XPRESS directly addresses critical challenges in computing of efficiency, scalability, and programmability through introspective methods of dynamic adaptive resource management and task scheduling.

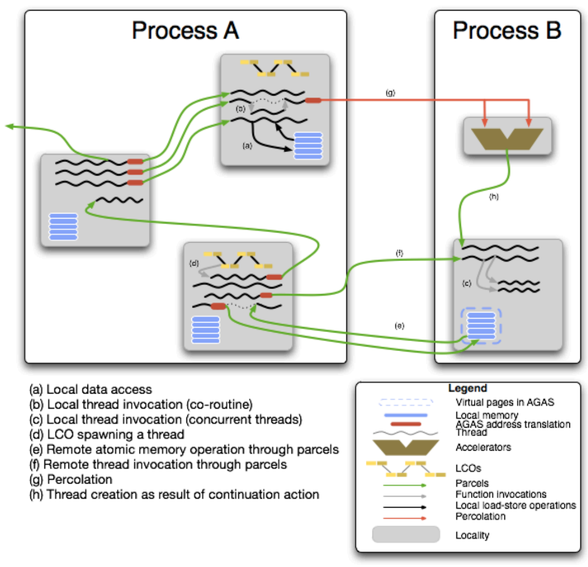
Major Tasks

- **Performance models & metrics** – provide parameters and their mutual sensitivities to guide co-design and quantify operational behavior
- **ParalleX execution model** – guiding principles for co-design of components of OpenX software stack
- **OpenX software architecture** – a conceptual framework for the co-design and interoperability of proof-of-concept XPRESS software stack including the RIOS interface protocol specification between the operating system and runtime system
- **LXK operating system** – Lightweight kernel operating system for order constant scalability and low/no noise to manage resources
- **HPX runtime system** – support of application dynamic adaptive resource management, task scheduling, and introspective control policies
- **XPI advanced programming model** – intermediate form and low-level (readable) programming interface reflecting the ParalleX model, providing a target for source-to-source high level parallel language translation, and supporting early direct programming experimentation and measurement
- **Legacy application mitigation** – ensuring seamless transition of legacy codes and programming methods to the future generation of ParalleX based exascale systems
- **Experiments and evaluation** – Critical to determining degree of effectiveness and likelihood of ultimate success as well as guiding corrective design changes to achieve DOE objectives
- **Documentation** – as well as reporting to DOE X-stack program management, to provide early adopters with sufficient information to apply prototype programming and execution environment

OpenX Software Architecture



ParalleX Execution Model



Goal: develop conceptual foundation to dramatically increase efficiency and scalability through dynamic resource management and task scheduling, and exploitation of new sources of parallelism

Means: An execution model to provide the governing principles of computation to guide the system codesign and interoperability of software component layers and portability across system classes

Performance Strategy:

- Scalability through lightweight thread parallelism with powerful fine-grain synchronization mechanisms, global barrier elimination, and exploitation of intrinsic metadata parallelism
- Latency mitigation through parcel based message-driven computation for limiting the number of remote actions, redistribution of flow control, and lightweight context switching
- Overhead reduction through optimized thread control and semantics of synchronization for minimum work
- Contention amelioration through dynamic resource management

HPX Runtime System

Emphasis on

- Functionality: finding proper software architecture
- Performance: finding hotspots, contention points, reduce overheads, hide latencies, do analysis
- API: finding the minimal but complete set of required functions
- Driven by real applications (AMR, Contact, Graphs, CFD, GTC)

Main tasks

- Manage parallel execution, mitigate latencies, resolve contention, minimize overheads for application in highly dynamic application environments
- Manage data dependencies, allow for data driven execution
- Enable medium to fine grain parallelism

Implements

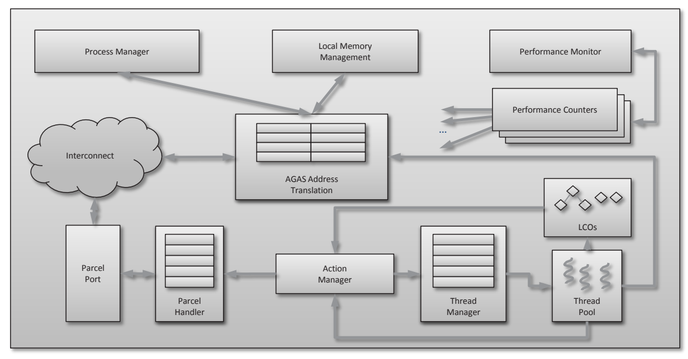
- Global active address space (AGAS), parcel transport, thread management, local control objects, parallel processes, performance counter framework

Exposes Interfaces

- Exposes a simple and usable interface to applications
- Tightly interacts with operating system to ensure high application throughput and scalability
- Guide software architecture design and modularization, and API design and interface definitions between modules

Enables

- Experimental verification of design, system architecture, and interfaces from very beginning
- Serves as backbone for early XPI implementation and early interoperability work (OpenMP, MPI)
- Basis for early application development and testing



Team

The XPRESS Project is led by Sandia National Laboratories (SNL) and engages a team of 8 institutions including: Indiana University (IU), University of North Carolina (UNC/RENCI), Oregon University (OU), University of Houston (UH), Louisiana State University (LSU), Oak Ridge National Laboratory (ORNL), and Lawrence Berkeley National Laboratory (LBNL).

Sandia National Laboratories · OAK RIDGE National Laboratory · LSU · BERKELEY LAB

http://xpress.sandia.gov