

OS/Runtime design challenges

- **OS/Runtime Interface**
 - Heterogeneity, Multiple address spaces
 - Black box, gray box, or white box
- **OS/Runtime customization**
 - Hypervisors
 - Library OS
 - Dynamic and/or automatic customization
- **Distribution of software stack**
 - Symmetric
 - Asymmetric
 - Side-core

Scalable Runtime / Operating System

- **Scheduling**

- Improve processor resource allocation management (QoS, Realtime)
- Scalability to manage larger number of cores (>32)
- Synchronization issues (Improve idle times between threads)
- Runtime Data/Communication dependences
- Thread affinity/migration over different PEs (heterogeneous)

- **Memory Management**

- Memory allocation over different address spaces
- Efficient data migration/mechanisms within different PEs and different address spaces (DMA, mailboxes, signals, atomics, ...)
- Global single address space

- **Interrupt/Exception Management**

- Scalable distribution between cores
- Event-driven scheduling

Ongoing efforts

- **HiPEAC Tools Clusters**

- Performance evaluation tools for heterogeneous multicore environments (UPC/BSC, FORTH-ICS, IBM Research)

- **OS Simulator (UPC)**

- Full system simulator: trace ANY relevant event (memory, syscalls, exceptions)
- Study the IMPACT of OS code over the architecture.
- Analyse/Visualize (Paraver)

- **I-cores hypervisor**

- Application-specific Oses for HPC/embedded systems
- Xen extensions for tick-less execution, core allocation, scalable interrupt processing
- rHype (IBM)

- **TPC runtime**

- Explicit parallel programming and memory hierarchy management
- OS/Runtime scheduling interface