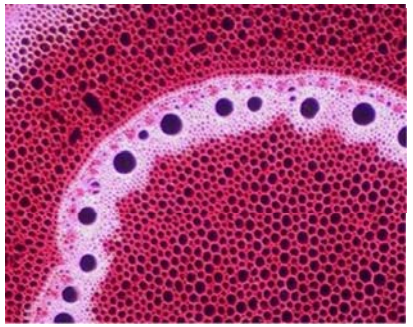
Georgia Tech, STI Workshop

Cellule: Virtualizing Cell for Lightweight Execution

Vishakha Gupta (Georgia Tech, IBM Research)
vishakha@cc.gatech.edu
Jimi Xenidis (IBM Research, Yorktown)
jimix@watson.ibm.com

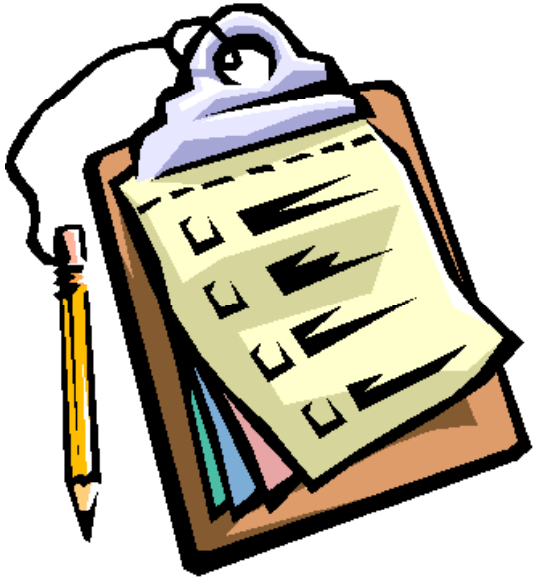
What is Cellule?

- Cellule - “a small cell”
- Use virtualization on Cell/B.E
  - Create a small high performance execution environment for SPE applications
- Each cellule becomes a true accelerator



Agenda

- Software components
- Cell architecture
- Remote IO
- Cellule architecture
- Plan of action

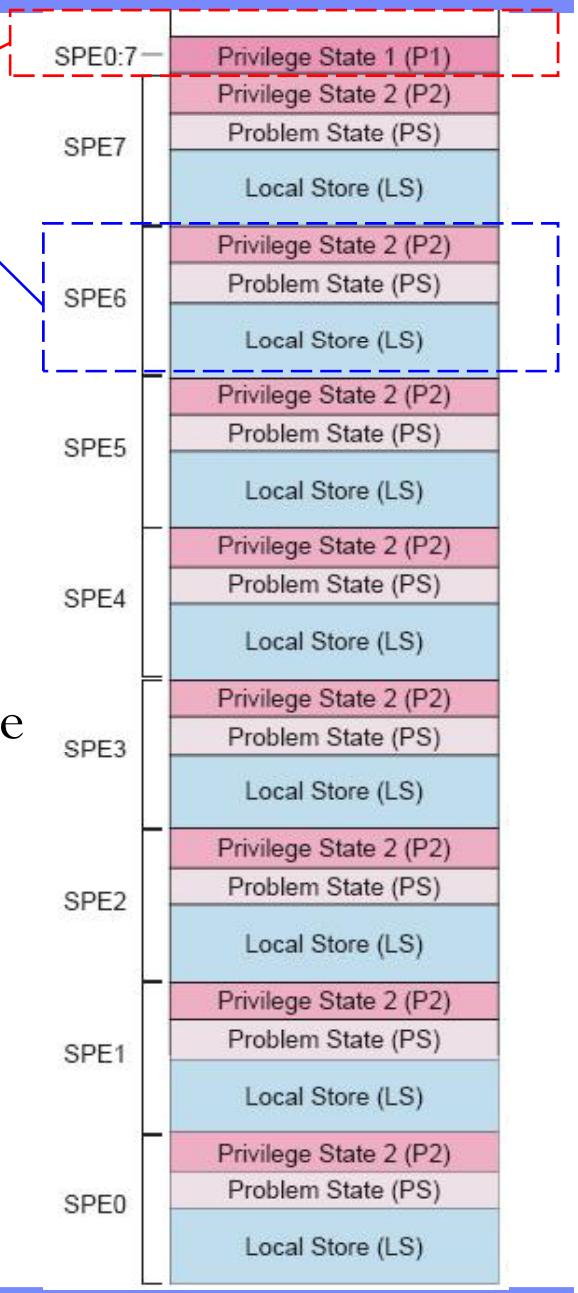


Software Components

- IBM Research Hypervisor (rHype)
  - Small, low-latency, modular Hypervisor
  - Specializes in partitioning accelerators, like SPEs
- Specialized Execution Environments (SEE)
  - Use Virtualization to create high performance customized environment for specific applications
- libSPE
  - Library for running SPE application from a PowerPC runtime
  - Tailored to the needs of application developers

Cell Architecture

- Cell performance driven by SPEs
  - PPE performance is poor
    - Suitable for management purposes
  - Decouple SPE operations from PPE operations
- Cell supports Virtualization
  - PPE side of SPE application run in Supervisor mode
    - Decreased interrupt latency
    - Intimate management of MMU operations
    - More MMU options (RMA, SW-TLB)
  - SPE Privilege 1 area accessible only to hypervisor
    - Partitions can have direct access to local store, problem and privilege 2 areas

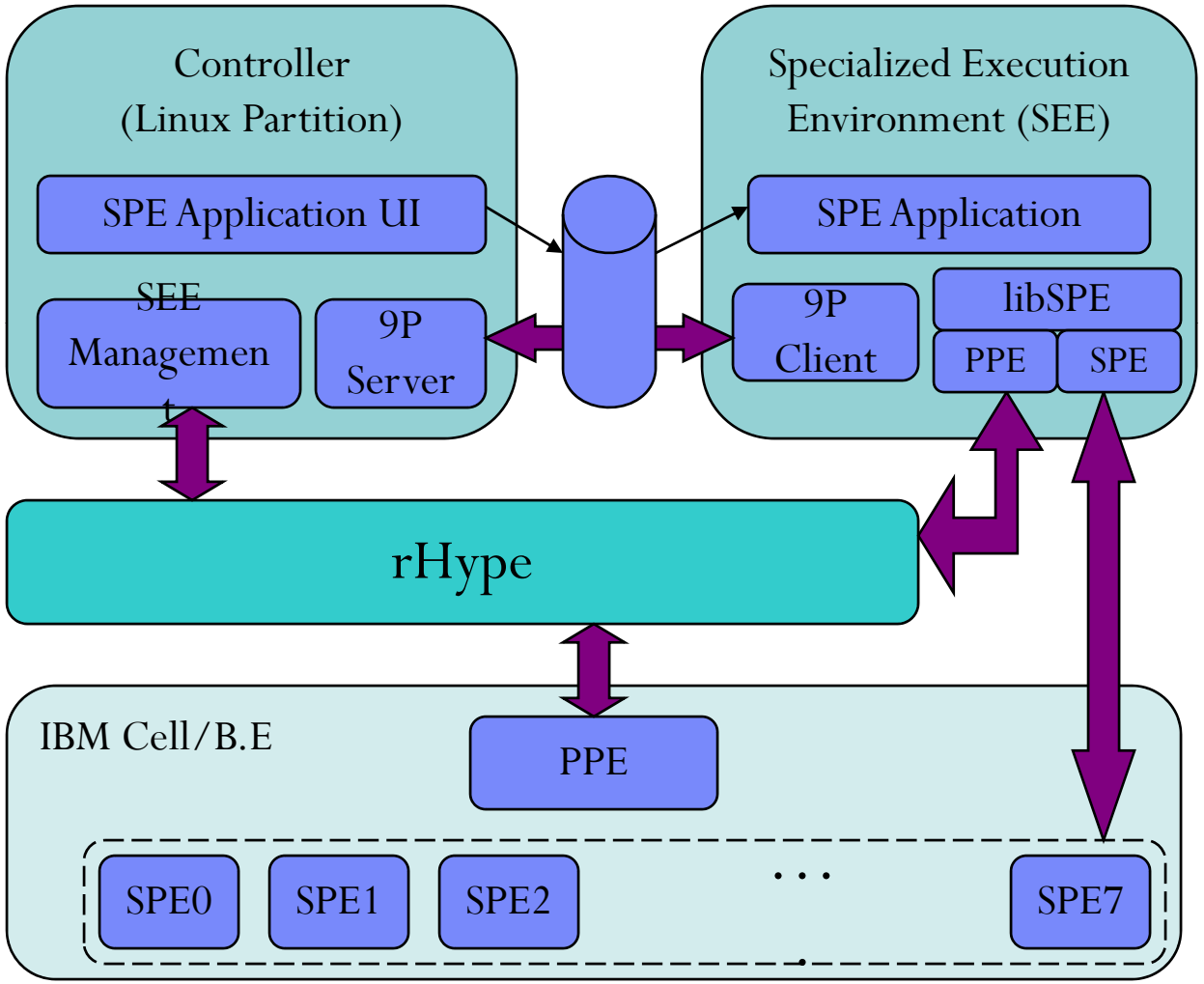


Remote IO

- General purpose OS only necessary for IO
  - Use simple remote IO protocol
    - 9P from Plan 9
    - Supports open/read/write and sockets directly
  - Can come from separate Linux Partition
    - Using shared memory transport for 9P
  - Can come from remote machine
    - Using IB, Ethernet or PCI-e

Cellule Architecture

- Controller seamlessly executes SPE application
  - Creates partition
  - Loads SPE app into partition
  - Partition provides libSPE to execute the application
- IO (file, console, network) happens through the 9P channel



Plan of Action

- Current internship (in progress)
  - Port rHype to Cell
  - Port simple benchmarks to SW stack
    - Simple controller
  - Port libSPE to SEE for use by any SPE application
  - Linux as Controller
    - Complete IO over 9P
- Future work
  - Remote acceleration – the “roadrunner” way
  - Performance of common benchmarks

Backup

References

- IBM Research Hypervisor (rHype), http://www.research.ibm.com/hypervisor/
- Specialized Execution Environments, http://domino.research.ibm.com/comm/research_people.nsf/pages/dgrove.vee2007.html
- libSPE, http://sourceforge.net/projects/libspe
- Plan 9 from Bell Labs, http://plan9.bell-labs.com/sys/doc/9.html

Design Issues

- Why 9P?
  - A simple and incredibly portable remote I/O protocol
  - It works as low as serial ports to shared memory
  - Creates familiar open/close/read/write interfaces
  - No driver, block, packet, tcp/ip stacks required
- Why not Xen?
  - Xen is designed for server platforms where high latency is a small issue
    - A millisecond delay can also be tolerated
  - Focuses on Virtualization rather than Partitioning
    - Usually one domain owns all physical devices
    - SPEs are designed to be partitioned
  - Does not use super/large pages