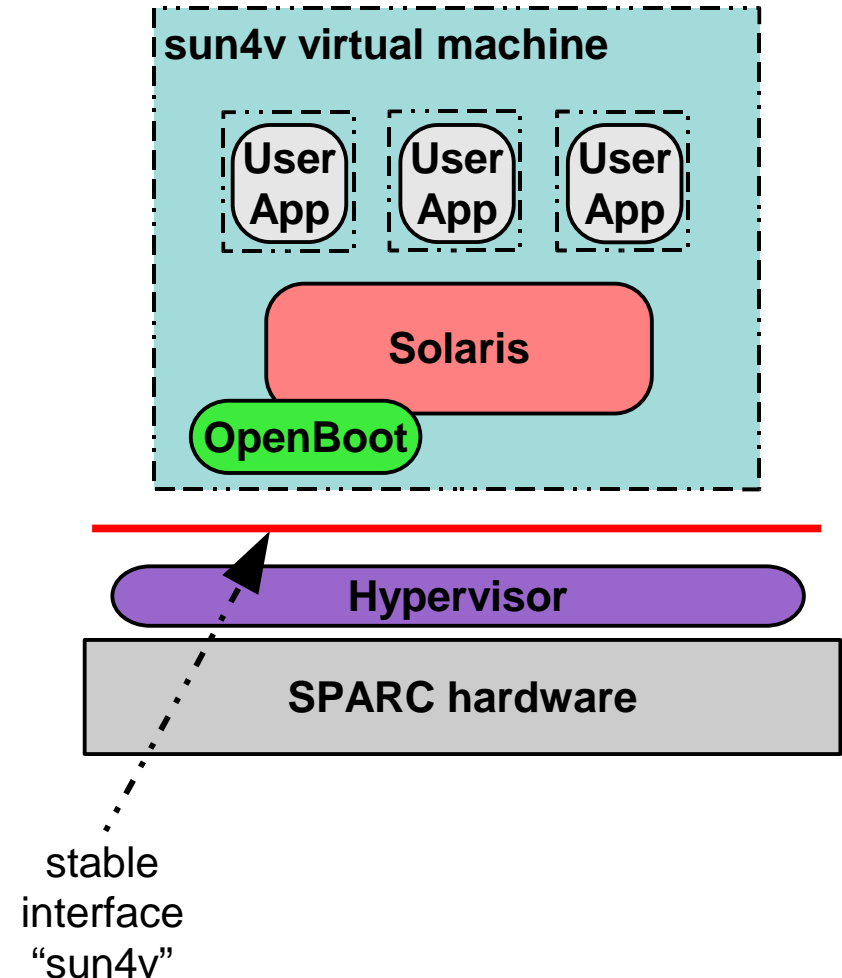


Your OS on the T1 Hypervisor

- Ashley Saulsbury
 - > Sun Microsystems Inc
 - > March 23, 2006
 - > blogs.sun.com/ash

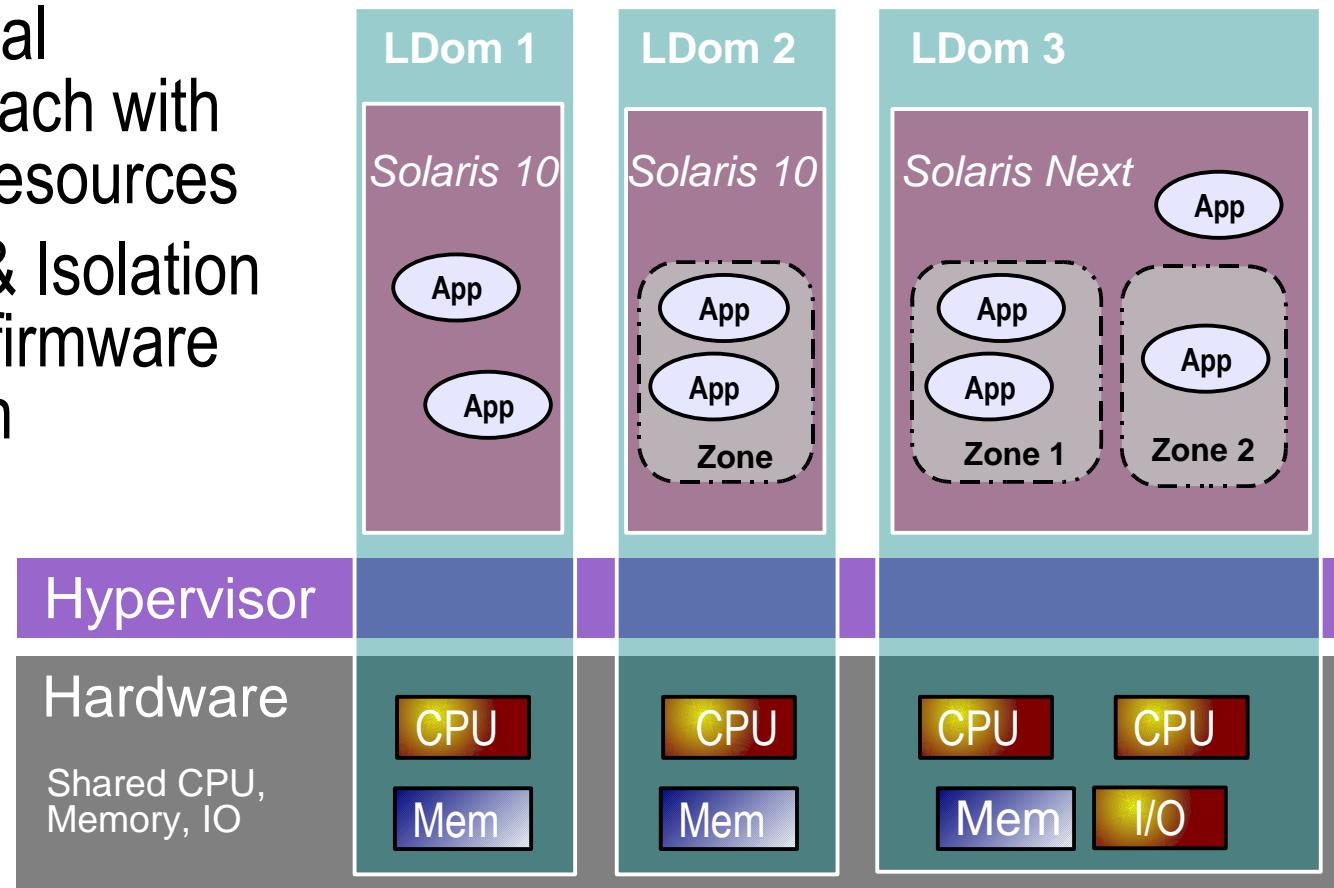
Virtual Machine for SPARC

- Thin software layer between OS and platform hardware
- Para-virtualised OS
- Hypervisor + sun4v interface
 - Virtualises machine HW and isolates OS from register-level
 - Delivered with platform not OS
 - Not itself an OS



Logical Domains

- Partitioning capability
 - > Create virtual machines each with sub-set of resources
 - > Protection & Isolation using HW+firmware combination

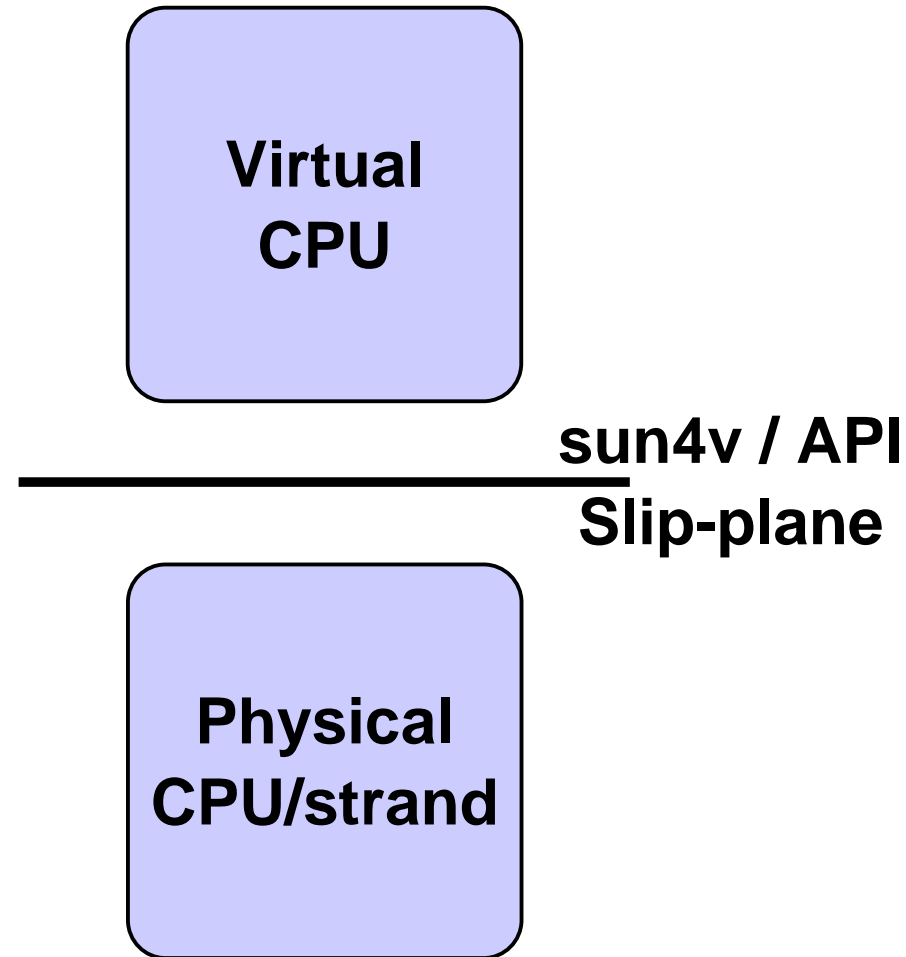


Topics

- CPU changes
- Memory management
- I/O
- Interrupts
 - > x-cpu & devices
- Error handling
- Machine Description
- Boot process

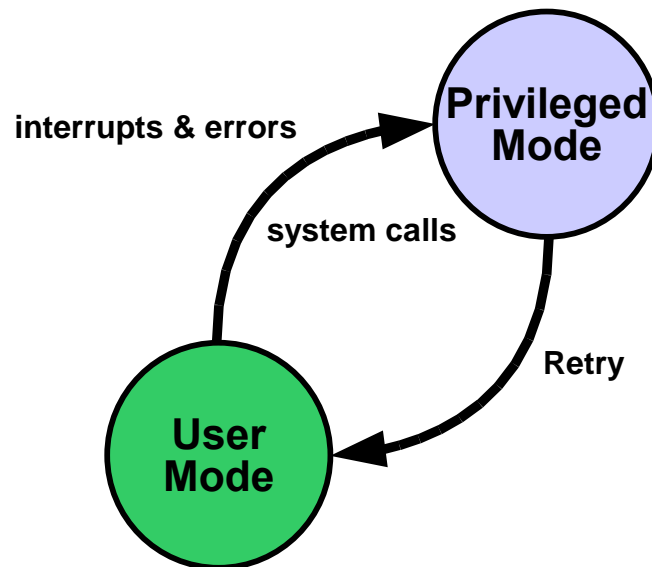
Basic Principles

- Ability to rebind virtual resources to physical components at any time
- Minimal state held in Hypervisor to describe guest OS
- Don't trust Guest OS at any time

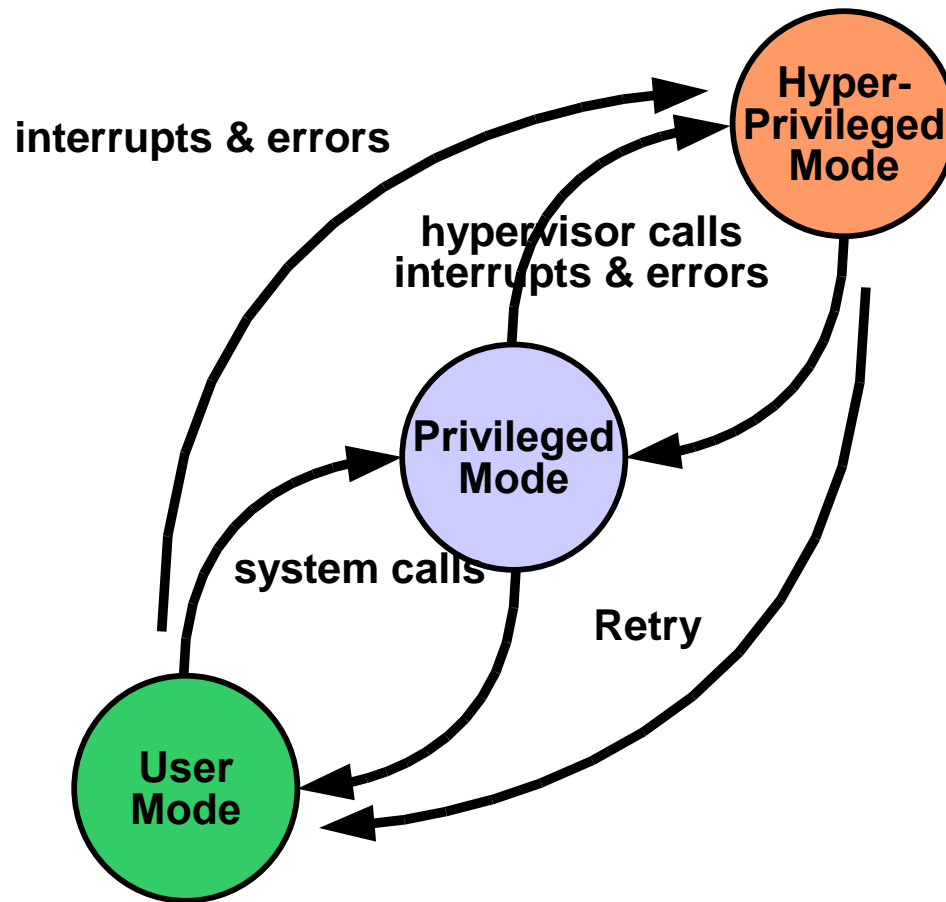


Legacy SPARC execution mode

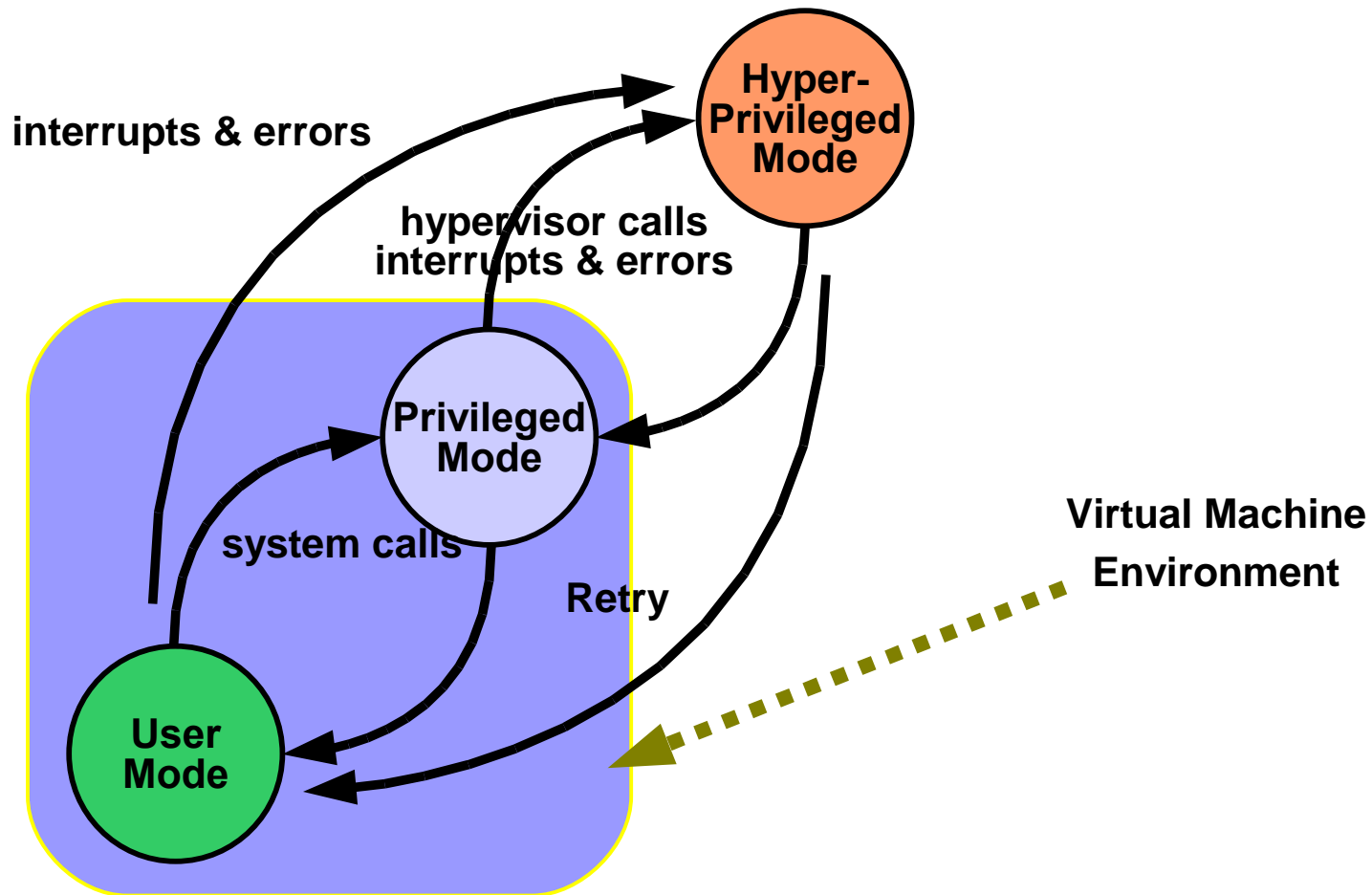
- Existing sun4u chips



New SPARC Execution mode



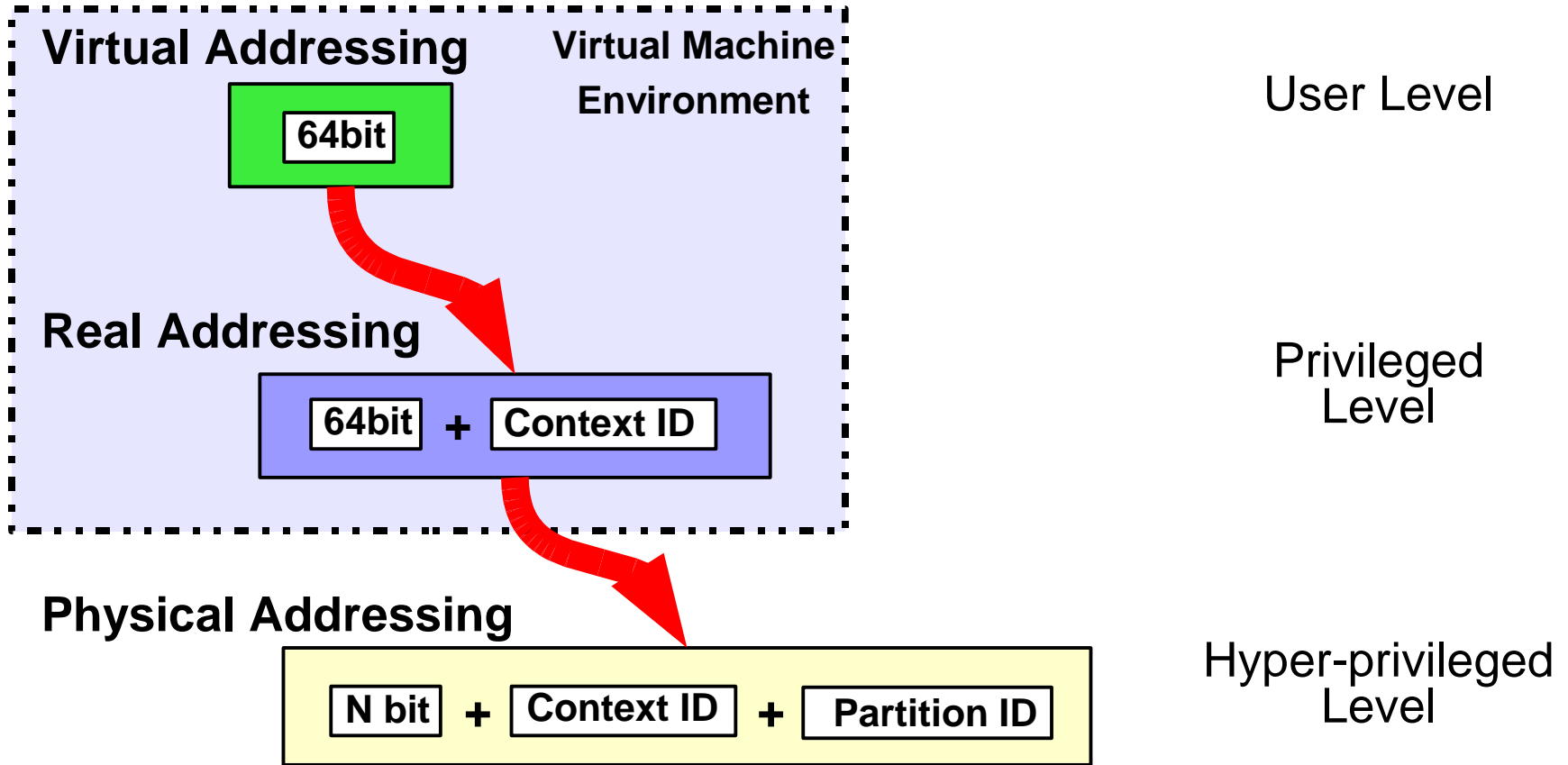
New SPARC Execution mode



Privileged mode constrained

- Close derivative of legacy privileged mode, but:
 - > No access to diagnostic registers
 - > No access to MMU control registers
 - > No access to interrupt control registers
 - > No access to I/O-MMU control registers
 - > All replaced by Hypervisor API calls
- UltraSPARCness remains with minor changes
 - > timer tick registers
 - > softint registers etc.
 - > trap-levels & global registers etc.
 - > register window spill/fill

Translation hierarchy



Translation management

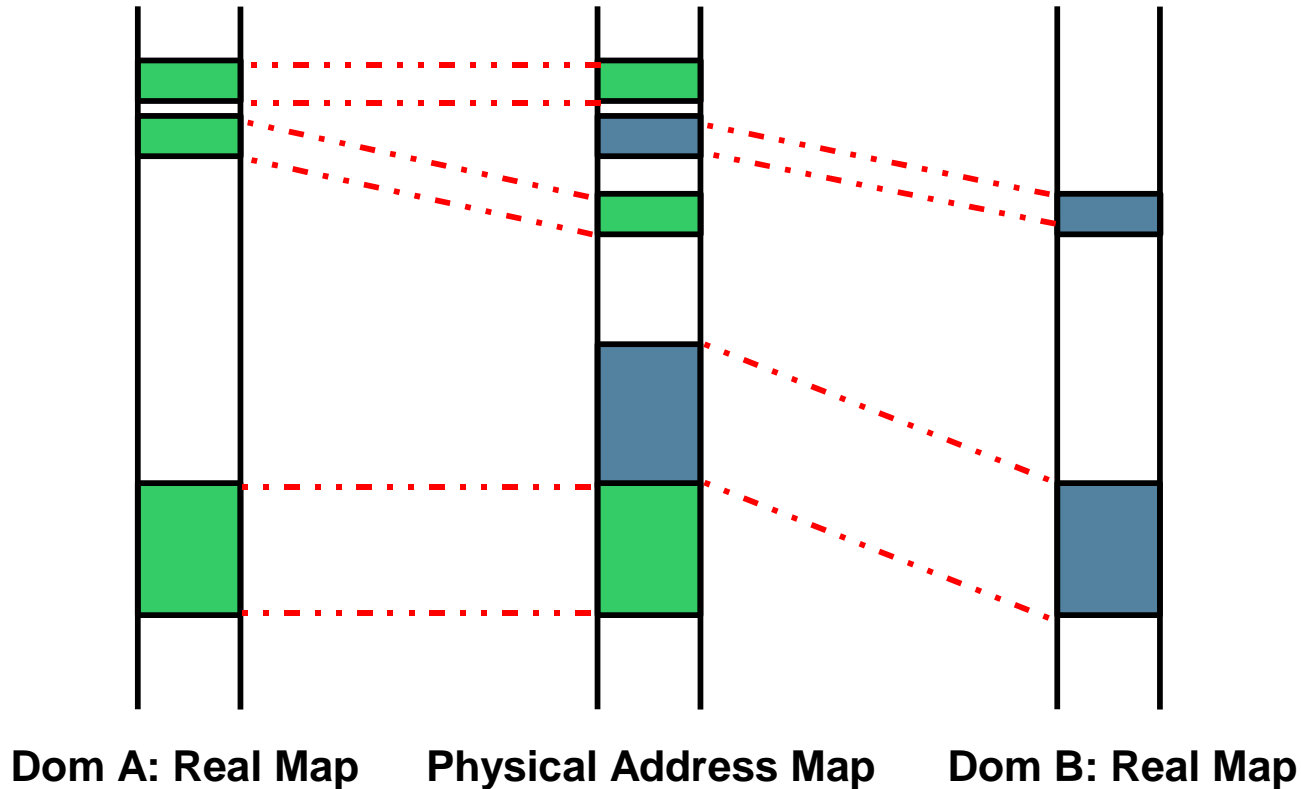
- Guest OS defines a “fault-status” area of memory for each vCPU
 - > Hypervisor fills in info for each MMU exception
- UltraSPARC does not use page-tables
 - > traditionally a software loaded TLB
- Hypervisor APIs to support direct software management of TLB entries
 - > map, demap
 - > Simple guests like OBP can use this

Translation Storage Buffers

- Guest OS managed cache of translations stored in memory
 - > Guest allocates memory for buffer
 - > Guest places translation mappings into buffer when needed
 - > Hypervisor fetches from this cache into TLBs
- Guest specifies virtual -> real mappings
 - > Hypervisor translates real->physical to load into TLB
 - > TLB holds virtual -> physical mappings
- Multiple TSBs used simultaneously for multiple page sizes and contexts

Address space control

- Hypervisor limits access to memory (and devices) - creating partitions (logical domains)



Virtual I/O devices

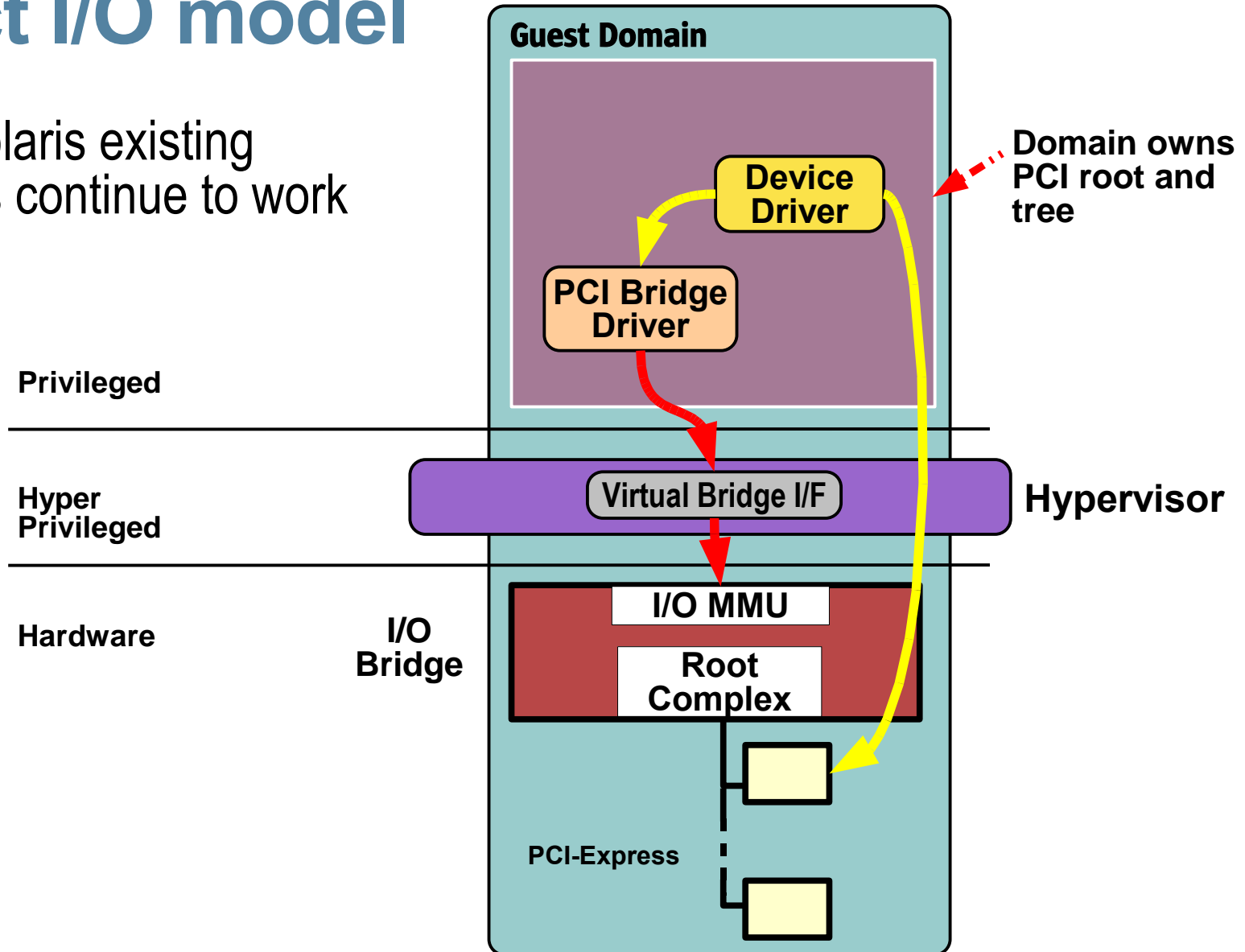
- Provided via Hypervisor
 - > e.g. Console - getchar / putchar API calls
 - > Hypervisor generates virtual interrupts

Physical I/O devices

- PCI-Express root complex mapped into real address space of guest domain
- Direct access to device registers
 - > OBP probes and configures bus and devices
- I/O Bridge and I/O MMU configuration virtualized by hypervisor APIs
 - > Ensures that I/O MMU translations are validated by hypervisor
 - > Device interrupts are virtualized for delivery

Direct I/O model

- For Solaris existing drivers continue to work

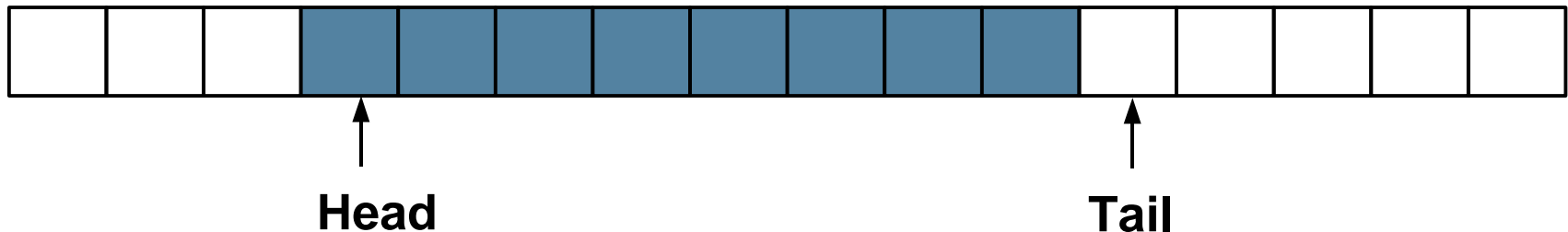


Interrupt and event delivery

- Device interrupts and error events need a mechanism to asynchronously cause an exception for a virtual CPU
- Typically also require some data to identify reason and source and notification
- How to do this in an abstract manner?
 - > What if the virtual CPU is not currently bound to a physical CPU?
 - > Can't block physical interrupt source until virtual CPU is rescheduled

Interrupts & CPU mondos

- Delivered to privileged mode via in-memory FIFO queues
 - > cpu-mondo, dev-mondo, resumable & non-resumable error queues
- 64 Byte entries carry cause information (interrupt numbers)
- Head and Tail offsets available as CPU registers to privileged code
 - > Tail manipulated by hypervisor, head by guest OS
 - > For either queue, head != tail causes trap



Q constraints

- Must be a 2^n number of entries, minimum of 2
 - > Entries always 64 bytes in size
 - > $(tail+1)\%size == head$ defines full state
- Must be aligned on a real address boundary identical to Q size
 - > Designed to make hardware mondo delivery easier
- May have queues defined for each virtual CPU
 - > dev_mondo queue must be sized for all possible interrupt sources
 - > dev_mondo queue may never contain more than one entry for same source
- Hypervisor API to send 64Byte mondo to CPU queue
 - > Used for CPU to CPU x-calls
 - > Queue may fill and sender's API call fails

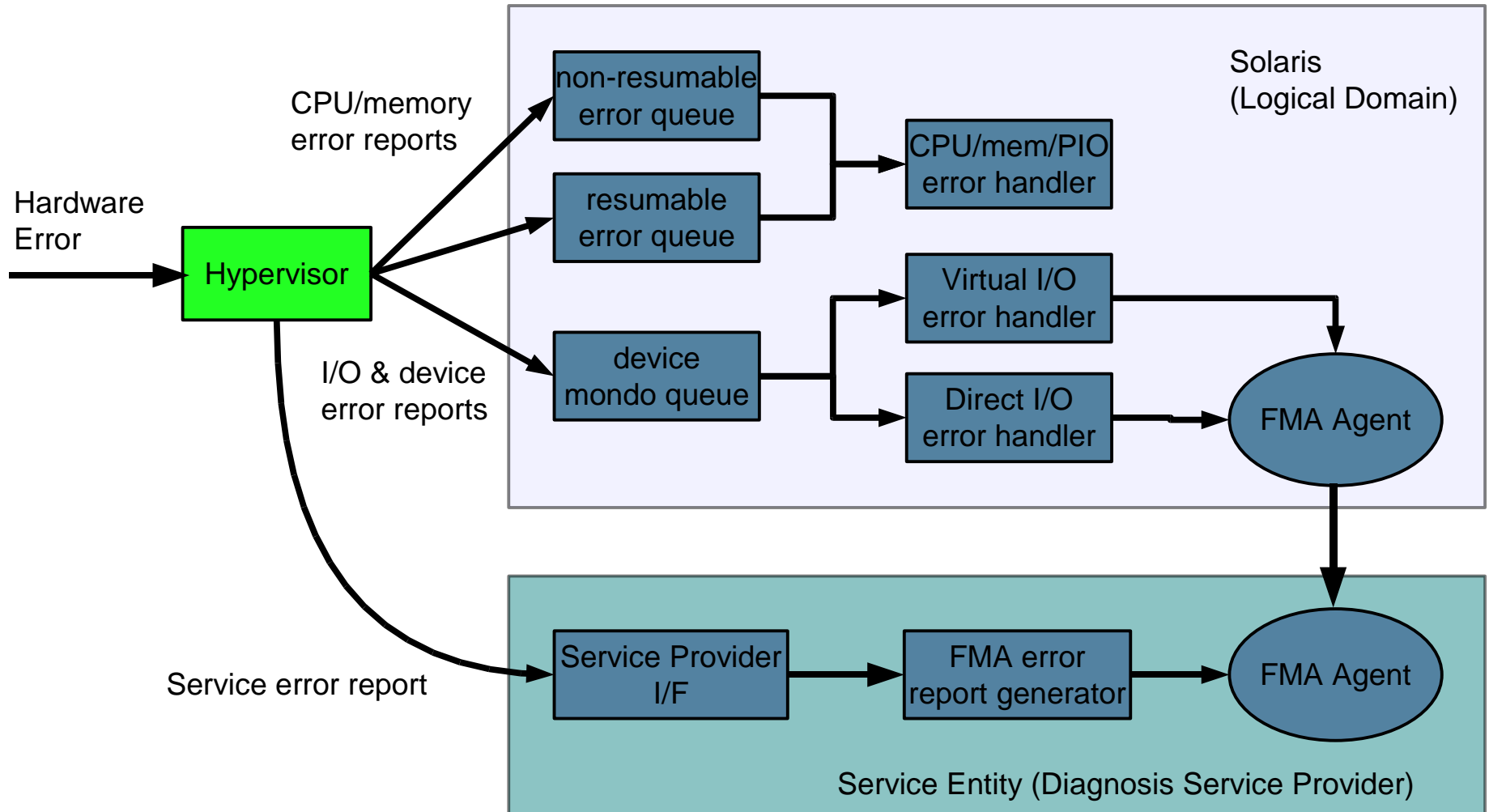
Error delivery philosophy

- Hypervisor handles and abstracts underlying hardware errors
- All errors logged on service processor
- Guest OSs are told about impact of error
 - > No point in informing guest about correctable errors
- Legacy OS should be able to run on new platform

Error handling

- Two simple classifications; “after handling the error can I ...”
 - > 1. Resume execution of what I was doing, or
 - > 2. Can't resume execution ... some policy to handle this
- Simplest error handlers;
 - > 1. Retry
 - > 2. Panic
- 2 in-memory queues associated with these types, similar to interrupts
- Queue entries contain error reports distilled by hypervisor
- Hypervisor creates reports and attempts to correct errors when possible

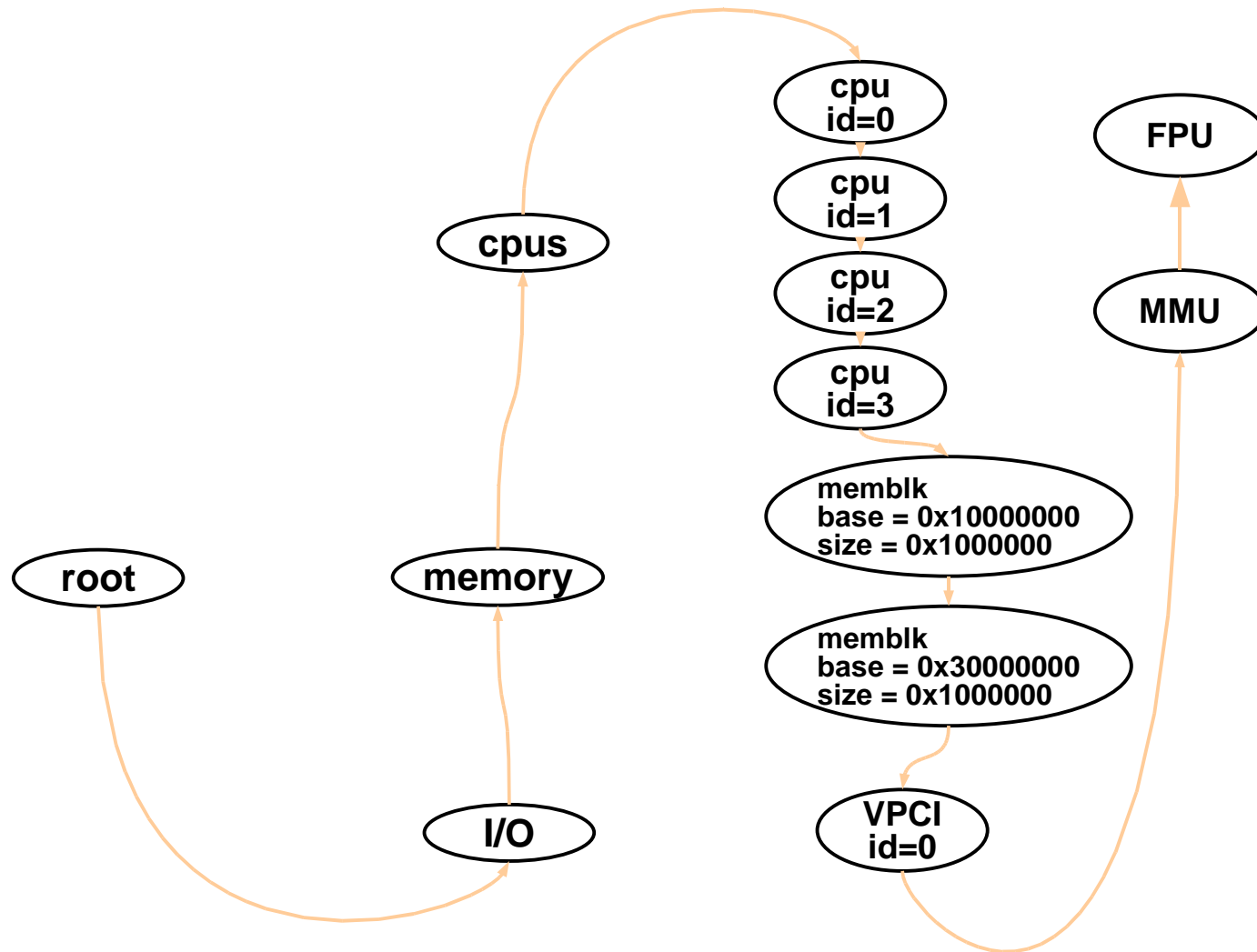
Error handling agents



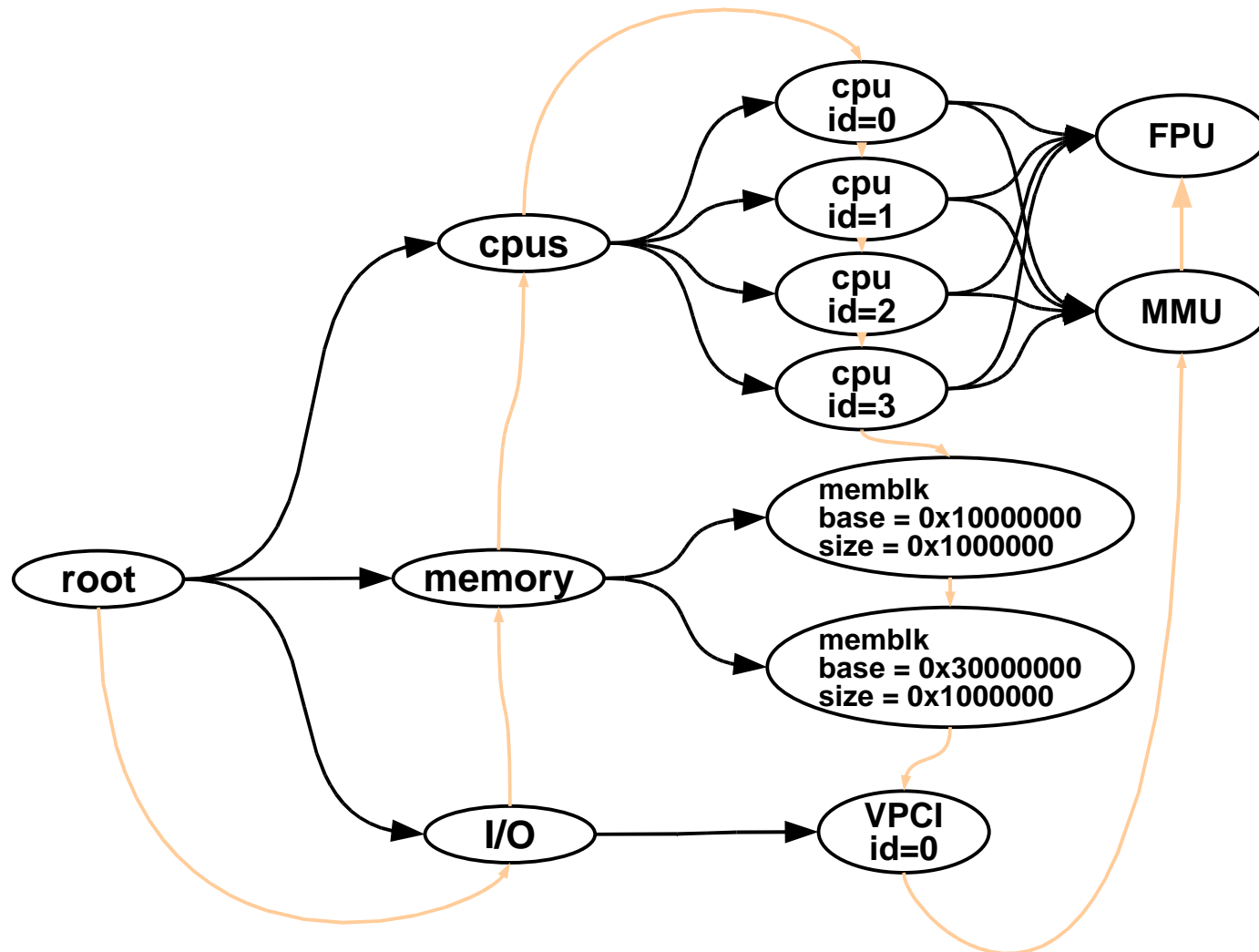
Machine description

- How the OS Inside a virtual machine finds its resources
- Trivial list of nodes that detail the contents of each domain
 - > CPUs, Blocks of memory, I/O devices, I/O Ports etc.
- Nodes are also inter-linked to form a DAG to convey more advanced information for guest OSs that care
 - > e.g. cache sharing, NUMA memory latencies etc.
- Key requirements;
 - > Very simple to parse by the simplest of guests
 - > Convey very complex information for guest OSs that care
 - > A guest need not understand all the information presented
 - > e.g. old OS running on a new platform

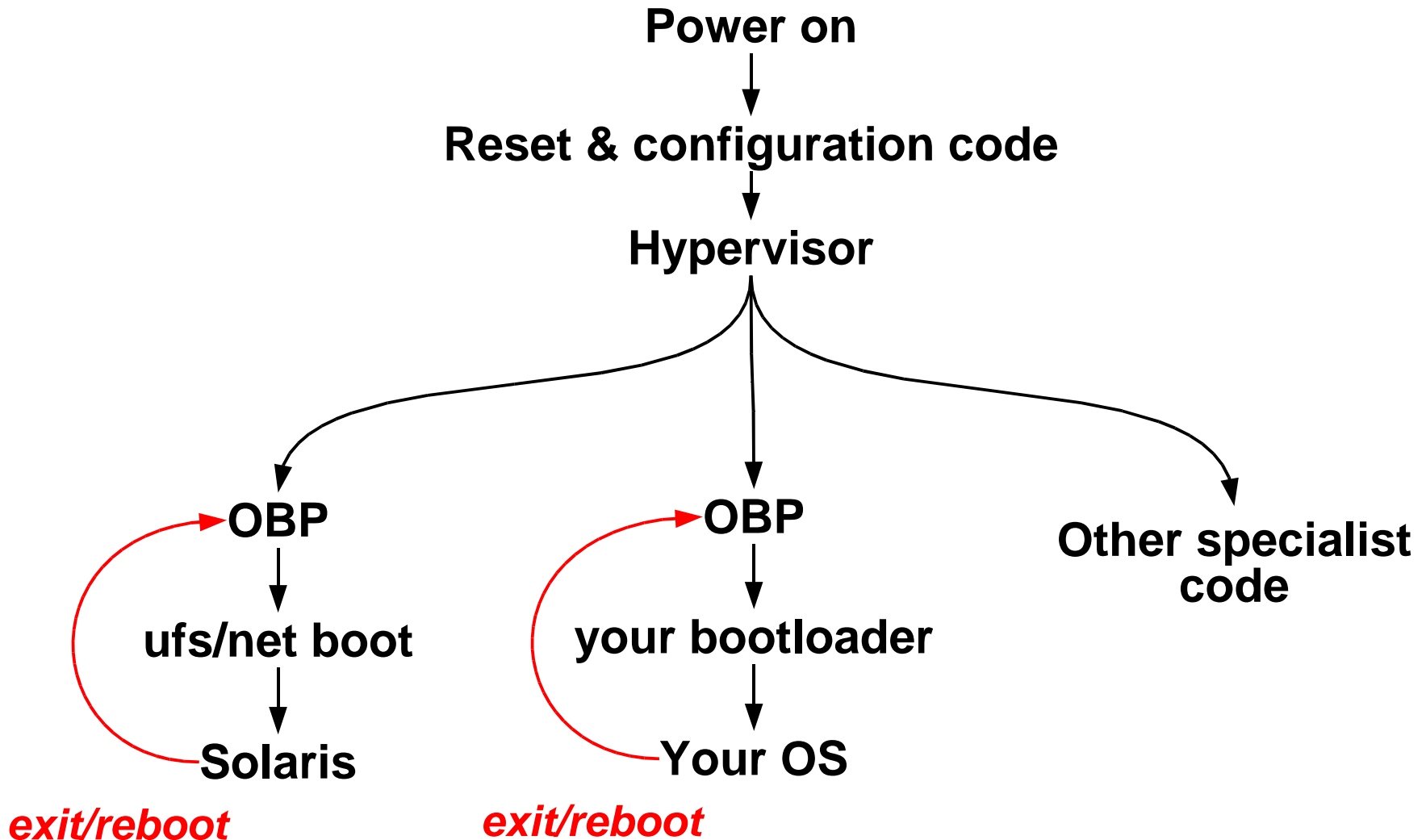
Simple list of nodes



Also arranged as a DAG



Boot process



Summary

- Specifications published;
 - > <http://www.opensparc.net>
- Software simulator available to assist with code development
 - > Provides level of code execution visibility not possible on actual hardware
- Contact alias:
 - > hypervisor@sun.com

- ashley.saulsbury@sun.com
- <http://blogs.sun.com/ash>